Personal Computer

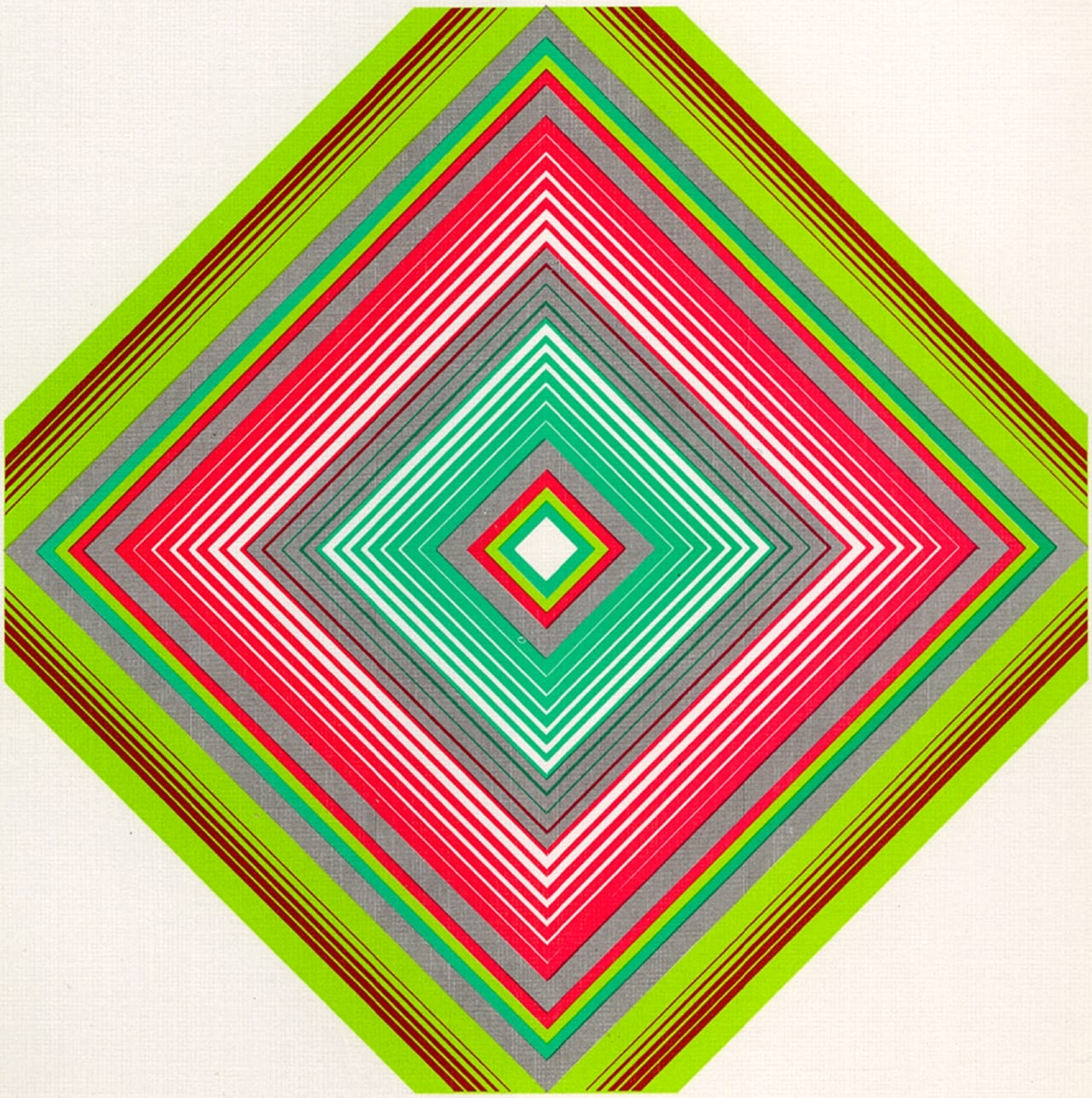
PC-8801 mkII

NEC

データファイル入門[2]

ナツメ社編集部＝編著

ナツメ社



Personal Computer
PC-8801mkII
NEC
データファイル入門[2]
ナツメ社編集部＝編著
ナツメ社

［本書を読まれる方へ］

⦿この本は、前に発行した
『PC-8801mkⅡデータファイル入門［1］』
の続編にあたるものです。
⦿したがって、ディスク操作の基本的な
コマンド、ステートメントなどについては、
いちいち解説を加えておりませんので、
これからディスクの勉強を始めようという方は、
まず、この本の前編をお読みください。
そして概論をマスターしたのち、この本に進めば、
落差が少ないので抵抗なく入れると思います。
⦿BASICをひととおり学び、
いざ実用化という段になって
必ず突き当たるのがデータの扱いです。
そこで、どうしてもマスターしなければ
ならないのがディスクです。
テープバージョンで仕事に役立てようとしても、
能率、スピード、正確さの点で
ディスクの足もとにも及びません。
一口にいうと、
まったく話にならないというのが当たっているようです。
⦿仕事に熱意をもっている方が、
せっかく購入したパソコンをなんとか
実用化したいと考えるのは人情です。
そんなあなたが、次のステップへ進めるよう
お手伝いしようというのがこの本です。
⦿きっと、あなたの仕事にプラスする
ヒントを発見できるでしょう。

cover design——nobuo nakagaki＋yoshifumi hayase

Personal Computer

PC-8801

NEC mkII

データファイル入門[2]

ナツメ社編集部＝編著

ナツメ社

この本を読まれる方へ

この本は、前に発行した

『PC-8801mkII　データファイル入門(1)』

の続編にあたるものです。

したがって、ディスク操作の基本的なコマンド、ステートメントなどについては、いちいち解説を加えておりませんので、これからディスクの勉強を始めようという方は、まず、この本の前編をお読みください。そして、概論をマスターしたのち、この本に進めば、それほど抵抗なく読めると思います。

本書、および前編とも、テーマには日常の卑近な例を取り上げていますので、ある程度ひたしみを持って進んでいただけるのではないかと考えております。ただ、ここで取り上げたプログラムが、そのままあなたの実務に適用できるということでは必ずしもなく、本書の目的が、むずかしいといわれているディスク操作の応用のしかたを具体例にそくしてやさしく解説する点にありますので、本書でのプログラムを変形したり、あるいはこれを参考に自分専用のシステムを作ったりするのは、あなた次第ということになります。

よく、「市販のビジネス・ソフトを買ったが、かゆいところに手がとどかない」、 つまり自分の仕事にぴったり合わない、といわれますが、それもそのはず、仕事の業務形態は各人各社まちまちなのですから、これを１つのソフトで充当しようというのが、どだい無理な話なのです。もし、これを使おうとするなら、自分の仕事内容をソフトに合わせなければならないことになり、本末転倒というものです。

しかし、そうはいっても、ワープロその他、市販のソフトでもかなり便利なものがあります。ですから、要は、市販のソフトにたよる部分、自分でプログラムを作って活用する部分というように、自分の仕事をある程度仕分けしてフレキシブルな態度で望んだほうがよいと思います。

まじめな方は、何がなんでも自分でプログラムを、と考えがちですが、できあいのソフトで自分の仕事に比較的適用できそうなものがあれば、これを購入して使ったほうが手間、労力の点でもずいぶん助かります。

ただ、こういったソフトを買う場合、自分で処理したい問題事項を

できるだけ列挙し、これをもって販売店に行き、そのソフトがどこまで希望をかなえてくれるか確認したほうがよいでしょう。

話が横道にそれましたが、ディスクはあなたの努力次第でずいぶん有効に使えるはずです。

容量も非常に大きくなっていますし、たとえば8インチのディスクに、この本でのフォーマットで宛て名を書き込むとすると、5000人以上は楽に保存できます。そして、こうしてデータファイルを作っておけば、あとはプログラム次第でどうにでも活用できるのです。

よく、マイコンショーなどではデモプログラムを走らせ、カラーグラフィックによる円グラフなどを表示させていますが、これは手間のかかるわりに効用は少なく、それよりも数値データをディスクに保存し、いろいろな分析方法を考え、プリンタで出力したほうが、どんなに役立つことかと思います。

そんなわけで、この本では、カラーモードは使わず、とにかく実用性を発揮できるように執筆・編集されています。

本書により、はやく次のステップへ進まれることを期待します。

編集部

もくじ

第1章

セクタのもう1つの使い方

1

1・1　ファイル内でのデータ設計

プログラミングに入る前に

私たちは、この本の前編

『PC-8801mkII　データファイル入門(1)』

で、シーケンシャルファイルおよびデータファイルについてのひととおりの知識を学んできました。

あとでふりかえってみますと、こんなことも添え書きしておけば、もっと理解の助けになったのではないか、と後悔される点もままありますが、一応「編集部の方針」つまり「パソコンの実用化」にそくして解説しましたので、80パーセントくらいはご理解いただけたのではないかと考えております。

ＢＡＳＩＣをひととおり学び、いざ実用化という段階にきて必ず突き当たるのがデータの扱いです。そこで、どうしてもマスターしなければならないのが、ＤＩＳＫ－ＢＡＳＩＣです。

テープバージョンで仕事に役立てようとしても、能率、スピード、正確さの点でディスクの足もとにも及びませんし、一口にいうと、まったく話にならない、というのが当たっているようです。

仕事に熱意をもっている人たちが、せっかく購入したパソコンをなんとか実用化したいと考えるのは人情です。パソコン活用が軌道にのったからといって決して楽ができるわけではありません。確かに、現在の事務処理その他については、ずいぶん能率化がはかれます。しかし、その先にまた次なる問題がひかえています。そして、この新しい問題とかテーマにチャレンジしていこうと意欲を燃やすところに、いわゆる「ヤル気のある人間」の「真実」があるといえますし、一社会人としての進歩が期待できるのです。

ある問題が解決してしまったら、もうそれでおしまいというのでは、これは「ごくあたり前の人間」です。

この本を読んで、何が何でもパソコンを実用化したいと考えているあなたがたは、おそらく大きな欲望をもっていらっしゃると思います。そして、この事実を前提に、筆者も筆を進めていきたいと思います。

プログラムも白紙の状態から作っていきますので、ずいぶん曲りくねった道をたどることも予想されますし、完成したからといって、いくつかバグもでてくるかもしれません。万一そんなことがありましたらご容赦ください。

プログラムで、いつも決まりきった仕事をする部分をサブルーチンといいましたね。プログラムでしたらルーチン化もわかりますが、私たちは、ぜひ「ルーチン化した人間」にならないようがんばろうではありませんか。

ずいぶん話が脱線してしまいましたが、結論は、実用化するならディスクを使わないと話にならないということです。

最初の本を読んでみなさんはどう感じられたかしれませんが、初めのうちは、ディスクというのはとにかくとっつきずらいものです。とくにデータがどんなふうに書き込まれているのか見えませんしね。しかし、試行錯誤を重ねているうちに徐々になれてくると思います。

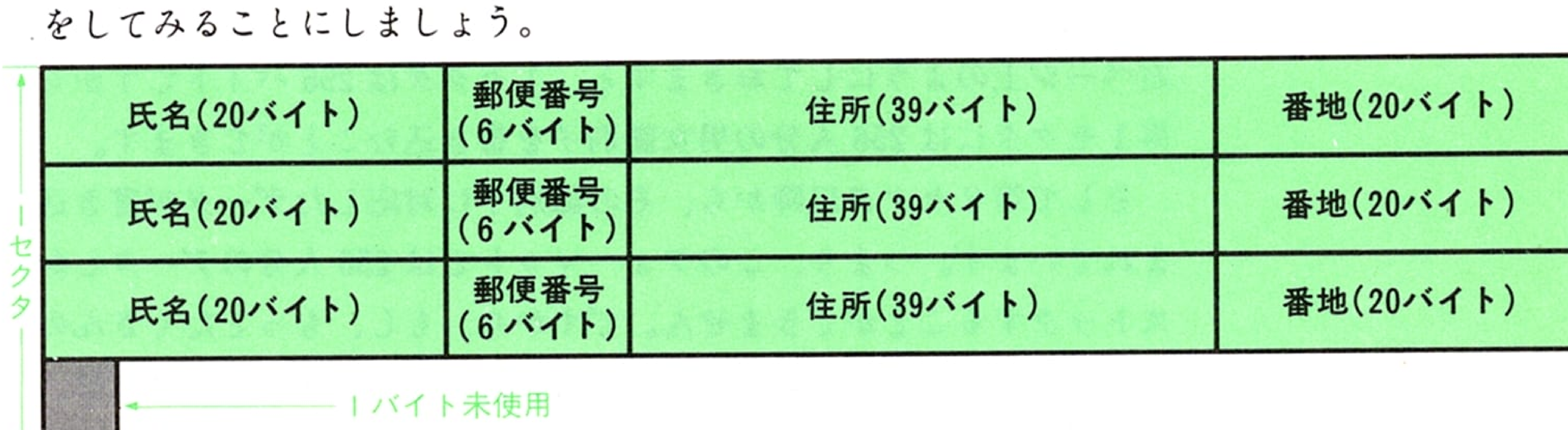
ディスクを扱うようになりますと、最初は**シーケンシャルファイル**から入ります。そこで、**ランダムファイル**よりシーケンシャルファイルのほうが理解しやすいというので、なんでもシーケンシャルファイルでかたづけようとする人がいます。しかし、これでは、仕事を型にはめてしまうようなもので、発展性がありません。なんとしても、ランダムファイルを自由に扱えるようになりたいものです。なぜなら、**ランダムファイルには非常にフレキシブルな面があります**し、慣れてしまえばそれほどむずかしくはないからです。

ちょっと違ったデータ設計

前著において私たちは、データファイルの１セクタ内に１レコードを書き込むことをまず学び、続いて１セクタ内に３レコードを割り当て、ディスクをさらに有効に利用することを知りました。

そのときのデータ設計は、およそ下の図のようでした。**氏名、郵便番号、住所、番地を１レコードとし、計３人分のデータを１セクタ内に書き込んだ**わけです。未使用は１バイトですから、大変効率的にディスクを使うことができました。

この章では、考え方をもう少し広げて、ちょっと違ったデータ設計をしてみることにしましょう。



この章で扱うテーマですが、話に連続性をもたせたいことと、そのほうが理解しやすいというメリットから、前回に引き続き同じ宛て名書きを選ぶことにします。

あなたがたが実用化したいというテーマはまちまちでも、データ設計の基本は共通していますので、この点をご理解ください。

私たちは前回は、3レコードを1セットとして第1セクタに書き込んでいました。しかし、ここでは、第1セクタはほかのデータを書き込むためにあけておき、氏名、郵便番号などのデータは第2セクタから書き込むことにします。

では、第1セクタを何のために利用するかといいますと、男性向けのダイレクトメール、女性向けのダイレクトメールを区別したいので、その性別を書き込むのです。

たとえば、1番目のレコードの人が男性であったら、第1セクタの1バイト目に「M」を、2番目の人が女性であったら、第1セクタの2バイト目に「F」を書き込む、というようにするのです。

こうしておくと、もし男性だけの宛て名書きが必要な場合、ファイルをOPENして、第1セクタのデータをチェックし、「M」という書き込みのあるものだけを抜きだし、次に、ここでわかった順位をもとに、第2セクタ以降から該当する位置をGETすれば、男性だけの宛て名を得ることができます。〈注〉

〈注〉この章でのデータの取り出し方が必ずしも合理的というわけではなく、あくまでセクタの使い方に重点があります。

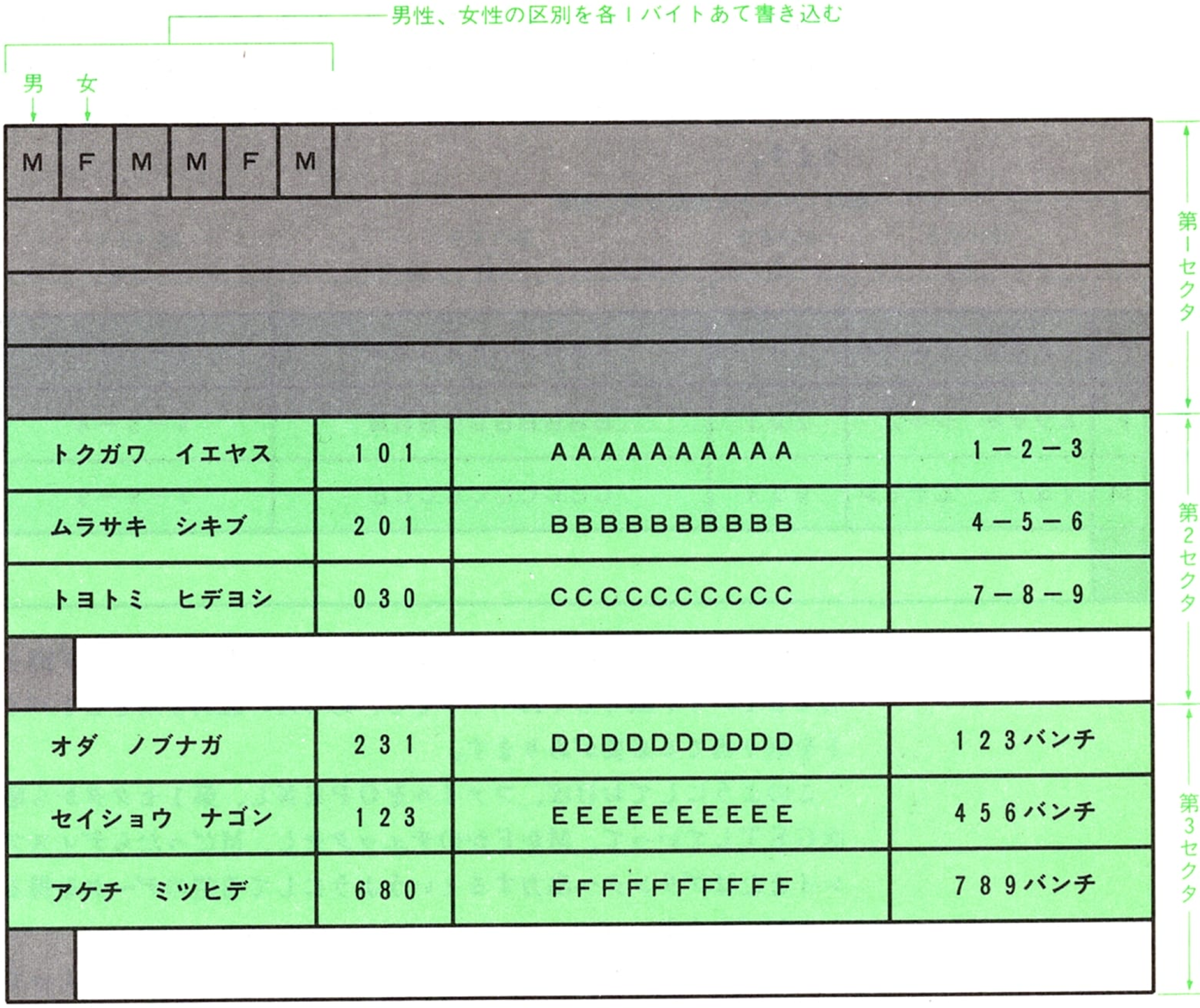
つまり、前回のファイル内でのデータ設計は、第1セクタから最終セクタまで、すべて同じフォーマットになっていましたが、ここでは第1セクタだけを異なるフォーマットで使用しようというわけです。

この様子を図に示すと、次ページの図のように表現できます。

ここで登場願った方々も、前回と同じようにだいぶ古い人たちですが、この原稿を書いている人が古いのでご容赦ください。とにかく理屈を理解していただくことが目的ですから、よろしくお願いします。

まあ、こんな話はさておき、このデータファイルのフォーマットを右ページ上のようにしておきますと、1セクタは256バイトですから、第1セクタには256人分の男女識別子を書き込むことができます。

そして第2セクタ以降から、その識別子に対応したデータが書き込まれています。つまり、このフォーマットでは256人分のデータしかストックすることができません。ですから、もし、もっとたくさんのデータ、たとえば1000人のデータを扱いたいというような場合には、「M」とか「F」を書き込むセクタをふやせばよいことになります。1000人のデータファイルを作るのでしたら、1000÷256≒3.9ですから、4



第4セクタ以降も第2、第3セクタと同じフォーマットで続く

セクタを識別子書き込みように確保しておけばよいことになります。

この場合、住所、氏名などのデータは第5セクタから書き込むことになります。

このように、ディスクの容量が許す範囲内で、どのようにも設計することができるのです。しかし、一度フォーマットを決めてしまうと、それを変更するには、古いデータファイルを新しいデータファイルに作りなおすプログラムを作ったりしなければなりませんので、かなりめんどうです。

ということは、**何年かはフォーマットを変更しないですむように、ある程度長期的な視野に立って考えたほうがよいと思います。**

さて、ここで1つだけ補足しておきますが、男性のデータだけを取り出したり女性のデータだけを取り出したりする場合、別に前ページ

の図のようなフォーマットにしなくとも、いままで私たちが学んできた形で考えることもできます。つまりその場合には、最初の第1セクタから次のようなフォーマットでデータファイルが作られることになります。

第1セクタ

前の1バイトを識別子用に使用	19バイト	6バイト	39バイト	20バイト
M	トクガワ　イエヤス	101	AAAAAAAAAA	1－2－3
F	ムラサキ　シキブ	201	BBBBBBBBBB	4－5－6
M	トヨトミ　ヒデヨシ	030	CCCCCCCCCC	7－8－9

この場合、フィールド設定のときに、氏名に対するフィールド割り当てを1バイト減らして19バイトとし、あらたに識別子用とし1バイトを割り当てる必要があります。

このようにしておけば、ファイルをＯＰＥＮし、第1セクタから順次ＧＥＴしていって、ＭかＦかのチェックをし、Ｍだったらディスプレイまたはプリンタへ出力するというようにして希望のデータを得ることができます。

処理時間もたいして変わらないと思いますし、このほうがかえって構造上簡単かもしれませんが、私たちにはいま「セクタ活用にもっと汎用性をもたせる」というテーマが与えられていますので、あえて、こちらは採用しないことにします。

また、この章では、セクタの新しい使い方を覚えていただくことが目的ですので、プログラムも簡単にし、この章で得た知識をもとに、次章でもう少し実用的なプログラム作りをこころみようと思います。

次のステップへの予備知識といったところですので、扱えるデータ数も256人までとします。

1・2　データの書き込み

それではまず、データ書き込み部分のプログラムから作ることにしますが、最初は初回のデータ書き込みだけを扱い、データを追加する場合は、このプログラムが完成したら、それを手直しして作っていくことにします。

プログラミングをする場合、最初からいっきに作り上げてしまおうとしないで、ある柱となるプログラムを作り、それに枝葉をつけていくというステップを踏んだほうが間違いも少ないですし、作りやすいようです。

前著の読者の方へ

この本の前編をお読みになった方もかなりいらっしゃると思いますので、少しだけ書き添えておきます。

この本では、前編から連続して入れるように、テーマもあまりへだたりのないものを選択しました。また、プログラム中で使用する変数名も、できるだけ前編に近い形で使うことにしました。この点、みなさんが理解しやすいように便宜をはかったつもりです。

また、前回と同じディスクをご使用になる方もいると思いますので、ファイル名は異なる形で命名するよう心がけました。ただ、テーマが類似しているため、多少似かよったファイル名になりますので、できたら新しいディスクにシステムディスクをバックアップしてご使用になるとよいと思います。

しかしこれは、前のディスクにまだアキがずいぶんあるとも考えられますので、どうしてもというわけではありません。

また前回と同様、プログラムファイルはドライブ1にセーブし、データファイルはドライブ2に書き込むことを原則としました。

プログラミングの途中で、バグがでたり、こんなことをつけ加えたらもっとよいのでは、といったこともでてくると思いますが、そんなときはみなさんと一緒に考え、修正していきたいと思います。

それからもう1つ、プログラムの中で1行のステートメントが長くなるものは、40桁で折り返してありますので、このことを前提にキーインしていってください。

初期設定

それではまず初期設定の部分から。

```
10 ' file name ==> DM01/1.bas
20 ' 1984.1.10
30 ' random data write
40 ' save "DM01/1.bas"
50 '
100 CONSOLE 0,25,0,0 : WIDTH 80,25
110 DEFINT A-Z
120 DIM NM$(3),PN$(3),AD$(3),BA$(3)
130 DIM SEX$(256)
```

これはもうおなじみのステートメントばかりですね。

行番号20の年月日は、プログラム作成日でしたね。もう1984年になってしまいました。早いものですね。

行番号40のステートメントの意味はわかりますね。そうです。このプログラムをセーブするときに使うのでした。わからない人は、前の本をもう一度ひっくり返してください。

画面モードの指定

このプログラムでの画面モードですが、前回左右40桁でやりましたので、今度は気分転換して80桁にしてみました。ちょっとだけ、コンピュータらしい気分にひたれるかもしれません。

変数の整数型宣言

行番号 110 で、A～Zで始まるすべての変数の整数宣言をしています。

配列宣言

行番号120、130は配列宣言です。ＮＭ＄が氏名用の、ＰＮ＄が郵便番号用の、ＡＤ＄は住所用、そしてＢＡ＄が番地用の配列変数です。

ＳＥＸ＄は何のための配列変数だかわかりますか？　ＳＥＸなどというと日本人には少々おかしな響きをもって聞こえがちですが、まともに考えてください。

前節で、第１セクタに男女を区別するチェッカーを書き込むとお話ししましたが、そのチェッカーのための配列です。

男性でしたら「M」を、女性でしたら「F」を書き込むわけですが、いずれにしても１文字書き込めばよいので１名について１バイトあれば十分です。そして１セクタは 256 バイトですから、第１セクタをまるまるチェッカー用として使うとすれば、 256 名分書き込めます。そして、 256 個のチェッカー用のフィールド設定は、配列を使って行ないますので、 256 の大きさの配列を宣言しておけばよいことになります。

ファイル内での書き込まれ方

さあ、初期設定ができましたので、使用するファイルのＯＰＥＮですが、その前に、男女の別、名前、住所などがランダムファイルの中でどのように書き込まれるか図示しておきましょう。

ここでのランダムファイルのディスクリプタは、

```
2:MAIL01.dat
```

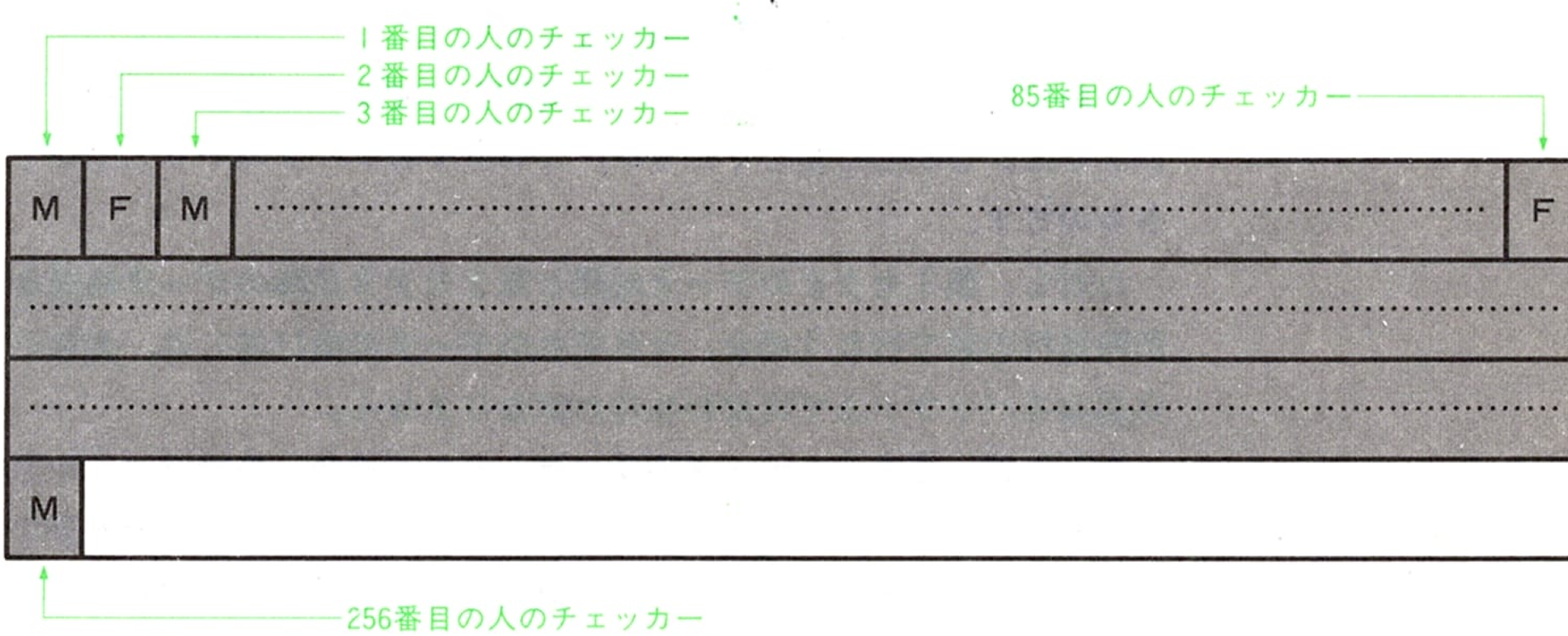
とします。

また、コード用のシーケンシャルファイルについては、

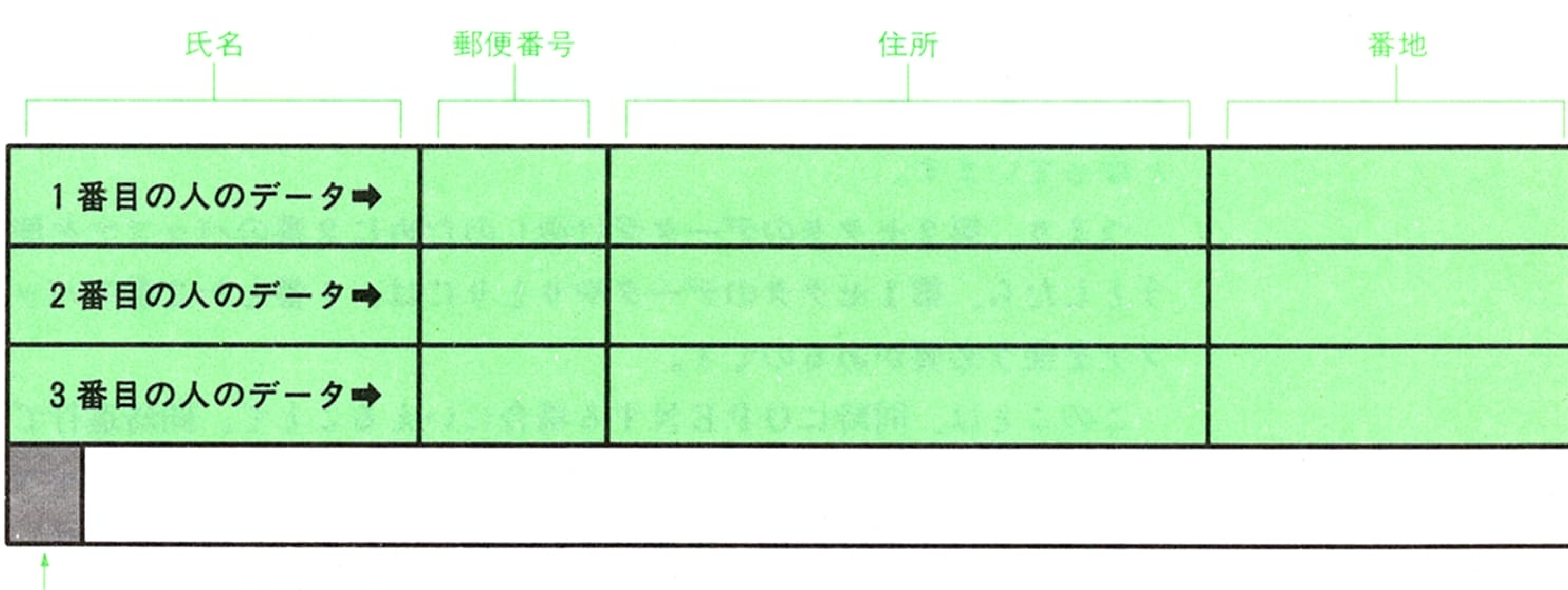
```
2:INDX01.dat
```

とします。

下の図をごらんください。これは、ランダムファイルの第１セクタ、



⬆ ランダムファイル「ＭＡＩＬ01.dat」内でのデータ格納状況（第１セクタ）



⬆ ランダムファイル「ＭＡＩＬ01.dat」内でのデータ格納状況（第２セクタ）

第2セクタでのデータの格納状況を図示したものです。

第1セクタは、256人分の性別チェッカー用として使用し、それに対応したデータは第2セクタ以降に書き込まれます。

バッファ上でのフィールド割り当て

前ページの図を見て、だいたいの様子をつかんでいただけたと思います。

前著では、第1セクタも第2セクタと同じフォーマットで書き込まれていましたね。ところが今度はまるきり違うフォーマットになっています。

といって、ファイル名が異なるわけではなく、どちらもまったく同じファイル名「MAIL01.dat」です。つまり、同一ファイル内に、まったく異なるフォーマットのデータが同居しているのです。これがランダムファイルの大変興味深いところです。

これで混乱が起こらないのかと、疑問をいだかれる方もいるかもしれません。しかしご安心ください。これで十分データのやりとりができるのです。

ただし、第1セクタのデータ処理と第2セクタ以降のデータ処理を同時進行の形で行なう場合、それぞれのデータの受け渡しは、お互いに異なるバッファ上で行なわなければなりません。

たとえば、どちらもデータの受け渡しをする場合、当然ファイルをOPENしなければなりません。それも同一のファイルをOPENするわけですが、使用するバッファを違えてOPENする必要があるのです。

バッファ

バッファというのはわかりますね。?

ファイルをOPENする書式で、「#1」とか「#2」というのをつけていました。このシャープのうしろの数字が使用するバッファの番号となっています。

つまり、第2セクタのデータ受け渡しのために2番のバッファを使うとしたら、第1セクタのデータやりとりには、1番とか3番のバッファを使う必要があるのです。

このことは、同時にOPENする場合にいえることで、同時進行でない場合、たとえば最初第2セクタのデータやりとりを2番のバッファを使って行っていたが、処理がすんだのでこれをCLOSEしてしまえば、今度は第1セクタのデータ処理のために、いま使っていた2番のバッファを使うことができるのです。

ちょっとややこしいかもしれませんが、要するに、同一のファイル

を異なるフォーマットで同時にＯＰＥＮしてフィールドを設定する場合、それぞれデータの受け渡しには異なるバッファを使用し、あるバッファを一度ＣＬＯＳＥしてしまえば、そのバッファはまったく開放され、どんなファイルにも使用できるのです。

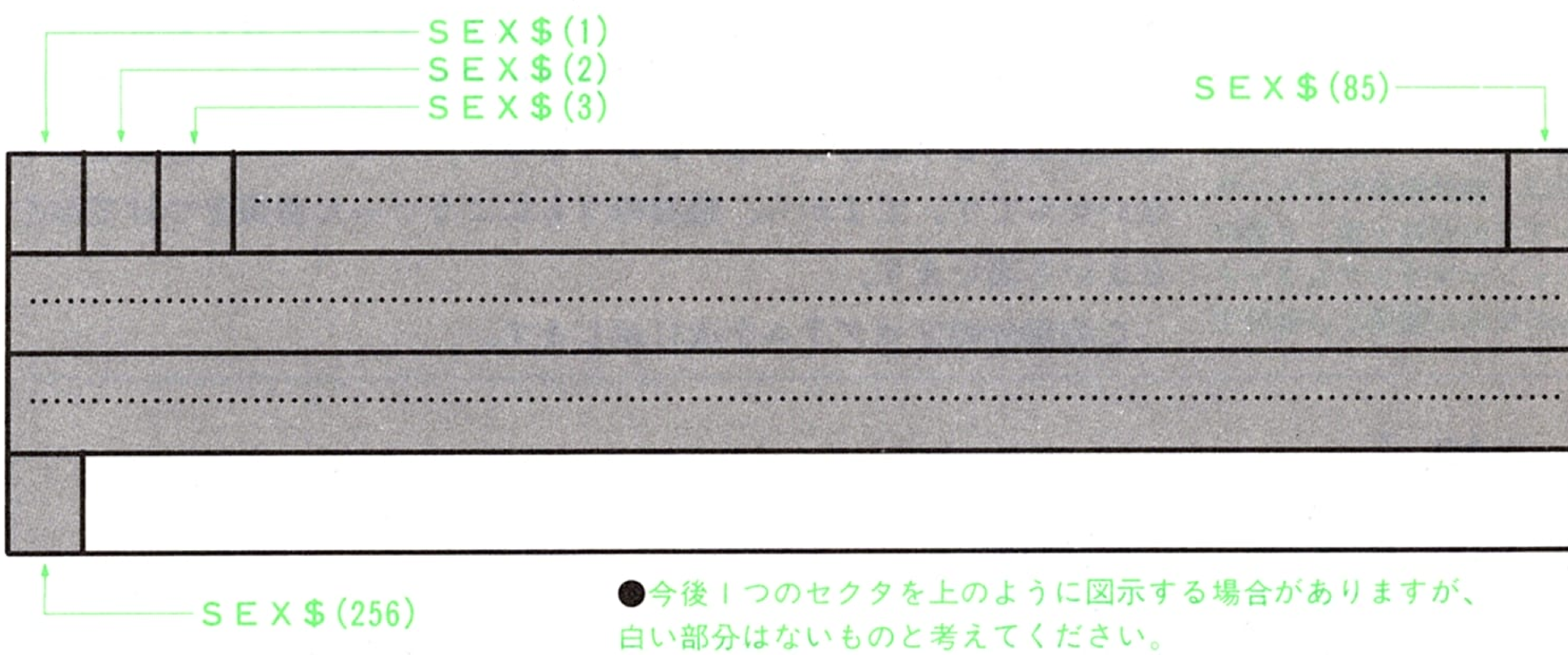
この辺は、理屈よりも実践をしていったほうが理解しやすいかもしれません。説明はこれまでにして、先へ進むことにしましょう。

さて、バッファ上でのフィールド割り当てですが、これも図に示すことにしましょう。

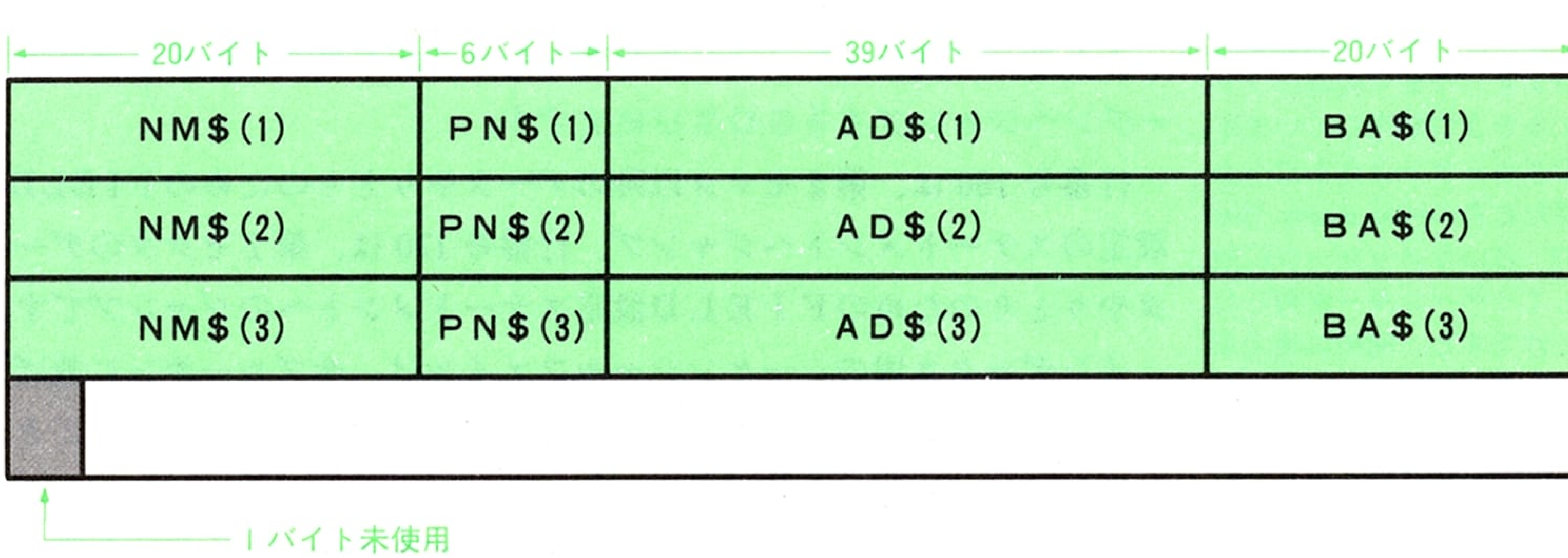
第１セクタのデータ処理には２番のバッファを、第２セクタ以降のデータ処理には１番のバッファを使うことにします。

セクタ番号とバッファ番号を対応させればよかったのですが、ご容赦ください。

次の図は、各変数のバッファ上でのフィールド割り当てを示したものです。



↑２番バッファでのフィールド設定



↑１番バッファでのフィールド設定

このプログラムでのフィールド設定、いかがですか？　なんとなく理解していただけたのではないかと思います。

それでは、いよいよファイルをＯＰＥＮし、ＦＩＥＬＤ文を作っておくことにしましょう。

ファイルをOPENする

このプログラムでは、ランダムファイルのＯＰＥＮはサブルーチン化しておくことにします。ただし、シーケンシャルファイルはサブルーチンとせず、使用するたびにＯＰＥＮステートメントを入れることにします。

サブルーチン化するのはなぜかといいますと、このプログラムではあまり意味をもちませんが、あなたがたがこれから徐々に複雑なプログラムを組んでいくようになりますと、ファイルのＯＰＥＮとＣＬＯＳＥをかなりひんぱんに行なうようになり、そのつどＯＰＥＮステートメントを入れていたら、それだけプログラムが長くなり、メモリ効率の点であまり好ましくないばかりでなく、プログラムの体裁もよくないからです。ですから、最初のうちにこういった習慣をつけておくとよいと思います。

この部分のプログラムを次に示します。

●下のリストの160、170行では、飛び先のサブルーチンでそれぞれ１番、２番のバッファをランダムファイルのために開放し、180行では３番バッファをシーケンシャルファイルに開放しています。つまり同時にファイルを３つ使用しています。ですからディスクを立ち上げたときのHow many files? というメッセージに対してリターンキーを押しただけですと、同時に使えるバッファ数は「2」に設定されますから、このプログラムを実行したときエラーになります。

```
140 '
150 CLS
160 GOSUB *OPNF01
170 GOSUB *OPNF02
180 OPEN "2:INDX01.dat" FOR OUTPUT AS #3
```

行番号 150 はおなじみの画面クリアです。

そして、行番号160、170がランダムファイルをＯＰＥＮするサブルーチンへジャンプするＧＯＳＵＢ文です。

行番号 160 は、第２セクタ以降のデータやりとりのためのＦＩＥＬＤ設定のステートメントへジャンプ、行番号 170 は、第１セクタのデータやりとりのためのＦＩＥＬＤ設定ステートメントへのジャンプです。

インデックス用のシーケンシャルファイルは、サブルーチンに飛ばしてＯＰＥＮせずに、行番号 180 でこれを行なっています。このことは先ほど説明しましたね。

さあ、そうしましたら、順序として、飛び先のサブルーチンを完成させてしまいましょう。

フィールドの多重定義

まず、順序が逆になりますが、第２セクタ以降のデータの出し入れをするフィールドから設定しておきましょう。

ここでは、氏名、郵便番号、住所、番地用の変数のフィールドを、それぞれ３人分について割り当てるわけですが、配列変数を使って簡潔な形に仕上げたいと思います。

この方法については前著に述べましたのですでにおわかりの方もいると思いますが、フィールド割り当ての様子を図示することによって、もう少し補足しておきたいと思います。

最初にプログラムを示します。

```
1000 '
1010 *OPNF01
1020 OPEN "2:MAIL01.dat" AS #1
1030 FOR I=1 TO 3
1040   FIELD #1,85*(I-1) AS DUMMY1$,20 A
S NM$(I),6 AS PN$(I),39 AS AD$(I),20 AS 
BA$(I)
1050 NEXT I
1060 RETURN
```

スペース１つ

40桁で折り返しています。みなさんは続けて入力してください。

このプログラムは、前著の形とほぼ同じですので、ごらんになるとわかると思いますが、ちょっとうなずけないという方は、前著をもう一度ひっくり返して、ある程度理解した上で、この記事に戻ってきていただければと思います。

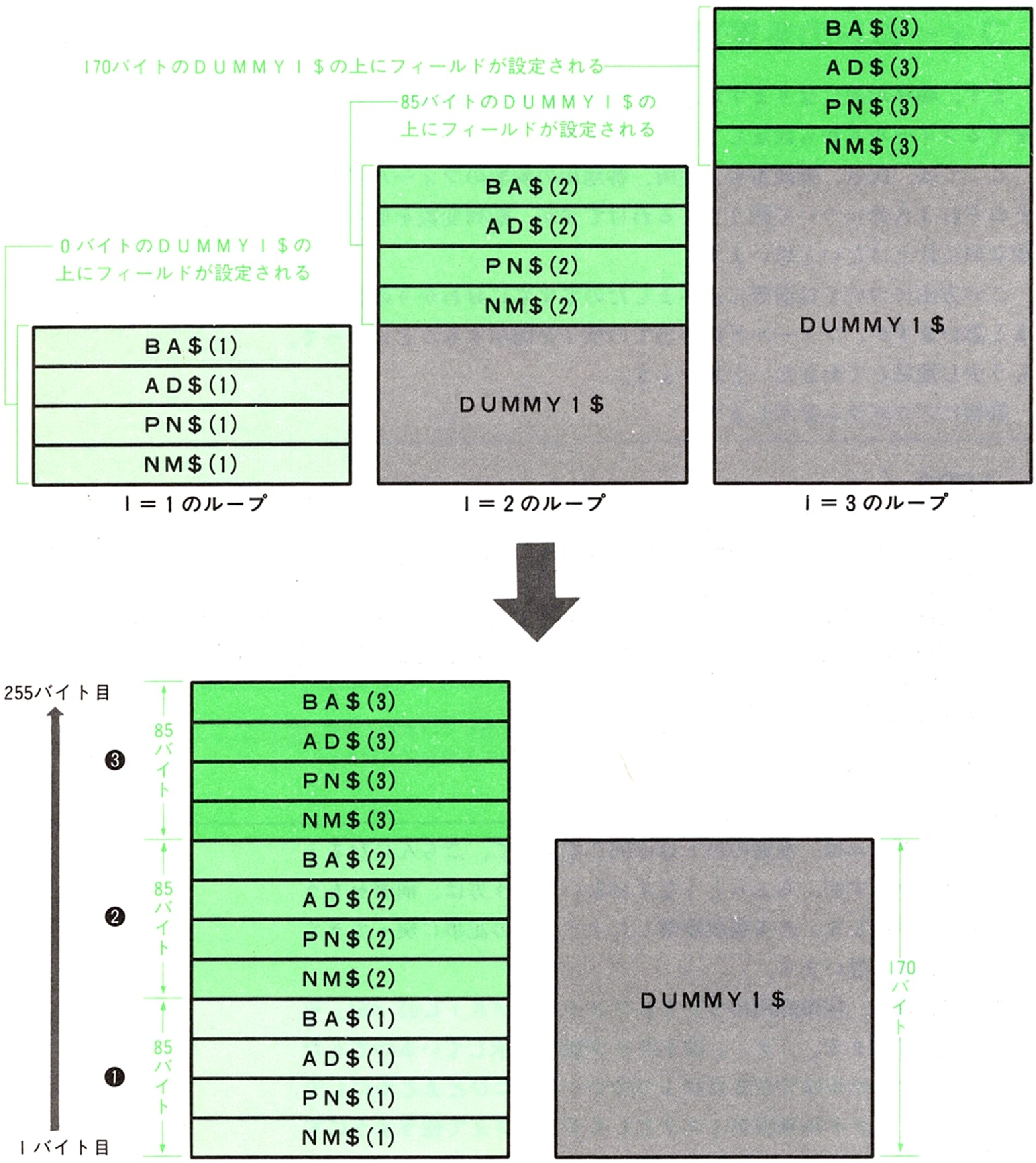
まず1020行で、住所録用のランダムファイル「ＭＡＩＬ01.dat」をＯＰＥＮしています。「２：」はドライブ番号を示しているのでしたね。データファイルはできるだけ１つのディスクにひとまとめにしておき、データファイル専用ディスクとしてドライブ２で使うようにすると大変便利です。

1030行から1050行までがフィールド設定のステートメントです。

1030行のＦＯＲ文と1050行のＮＥＸＴ文ではさみ、３回ループさせ、１回ループするたびに前のフィールドに重ねて定義しています。

この辺のプロセスが初めのうちはわかりにくいと思いますので、この様子を次ページに図示しました。

図を見ながら説明をお読みください。



●FOR～NEXT文で3回のループが終了すると、フィールド上でデータの受け渡しをできる変数のフィールドは、最終的には下側の図のようになります。それぞれの変数になわ張りが決められたわけです。このほかにもプログラム中ではいろいろな変数が使われますが、バッファ上でデータのやりとりに参入できるのは、上の図の変数のみということになります。DUMMY 1 $もダミーとはいえ、元気に生きているわけですが、私たちがGETさえしなければ混乱は起こらないことになります。

まず最初のループではＩ＝１ですから、ＤＵＭＭＹ１＄に対するフィールドは

85＊(Ｉ－１)＝85＊(１－１)＝０

となります。０ということ、つまり０バイトがＤＵＭＭＹ１＄のフィールド幅として割り当てられます。ということは、何もないということと同じですね。

そして、この何もないところに配列ＮＭ＄(１)、ＰＮ＄(１)、ＡＤ＄(１)、ＢＡ＄(１)のフィールド幅が20バイト、６バイト、39バイト、20バイトというように続けて設定されていきます。ですからこの段階では、ＤＵＭＭＹ１＄という変数は存在するのですが、フィールド幅は０ということになります。

ここで１つおことわりしておきますが、左ページの図ではＤＵＭＭＹ１＄以外の変数はどれも同じバイト数になっていますが、実際には異なるバイト数ですのでご了承ください。また、フィールドが設定される様子も、理解しやすいのでこのようにしました。

次は２回目のループ、Ｉ＝２の場合ですが、今度はＤＵＭＭＹ１＄に対するフィールドは、

85＊(Ｉ－１)＝85＊(２－１)＝85＊１＝85

となります。

つまり、85バイトがＤＵＭＭＹ１＄のフィールド幅として割り当てられ、その上にＮＭ＄(２)などのフィールドが設定されます。

ここでよく誤解しやすいのですが、ＤＵＭＭＹ１＄に85バイトが割り当てられたので、Ｉ＝１のループで設定されたＮＭ＄(１)～ＢＡ＄(１)のフィールド85バイトはなくなってしまうのではないかという心配がでてきますが、これはまったく支障ありません。

というのは、上のような形で一度フィールド幅が割り当てられれば、一種の権利が生じたようなもので、消えることはありません。

最後に３回目のループ、今度はＩ＝３ですから、ＤＵＭＭＹ１＄に対するフィールド幅は、

85＊(Ｉ－１)＝85＊(３－１)＝85＊２＝170

と、だいぶふくれ上がりました。２回目のループのときのＤＵＭＭＹ１＄の中味85は追い出されて、今度は170がその内容となりました。そして、この170バイトの上に、ＮＭ＄(３)～ＢＡ＄(３)のフィール

ドが割り当てられるのです。

これで、３回のループが終了し、フィールド設定ができたわけですが、最終的なテリトリーの構図はどうなったでしょうか。これを示すのが20ページの下の図です。

ファイルをＯＰＥＮしたときバッファ上で活躍できる変数名およびそのテリトリーは図に示す通りです。

たとえばＰＮ＄(１)でしたら、１セクタ内の21バイト目から６バイト、つまり25バイト目までが管轄になるわけです。あくまで管轄ですから、ＧＥＴ文、ＰＵＴ文でお呼びがかからなければ活躍のしようがありません。

一方、このプログラムでの処理に関係のないＤＵＭＭＹ１＄も図でわかるようにテリトリーを与えられています。このなわ張りは大きいですね。❶と❷を合わせた170バイトです。ですから、ＧＥＴ文でセクタをＧＥＴし、ＤＵＭＭＹ１＄を呼び出せば、１バイト目から170バイト目までのデータがＤＵＭＭＹ１＄に入ってしまいます。

たとえば、❶と❷の位置に氏名と住所などが書き込んであれば、２人分のデータをまるまるのみ込んでしまうことになります。このことは逆に、ＰＲＩＮＴ文を使ってＤＵＭＭＹ１＄の内容をディスプレイに表示させれば、２人分のデータをいっぺんに表示させることができる、ともいうことができます。

この場合にはどのデータも文字列(ストリング)ですからよいのですが、これが数値データなどですと、メチャクチャな値になってしまいます。

では次に、第１セクタのデータやりとりのための変数のフィールド割り当てです。

これもやはり、ＦＯＲ～ＮＥＸＴ文を使って、きれいに設定したいと思います。

行番号を2000からとして次のようにキーインします。

```
2000 '
2010 *OPNF02
2020 OPEN "2:MAIL01.dat" AS #2
2030 FOR I=1 TO 256
2040   FIELD #2,(I-1) AS DUMMY2$,1 AS SEX$(I)
2050 NEXT I
2060 RETURN
```

2020行はランダムファイルのＯＰＥＮです。

使用するバッファは２番を指定しています。1020行でＯＰＥＮしたファイル名と同じですから、もしここで１番バッファを指定してＲＵＮすると、次のようなエラーメッセージが現れてプログラムがストップします。

```
File already open in 2020
Ok
■
```

2020行で、すでにＯＰＥＮしてあるファイルをＯＰＥＮしようとした、というメッセージです。

プログラム実行中にこのようなエラーがでたら、あわてずに、まずダイレクトモードで「ＣＬＯＳＥ⏎」をキーインし、バッファ番号をなおしたりするとよいでしょう。

ここで１つ注意していただきたいことは、第１セクタのデータやりとりは常に２番バッファを使うと決められているわけではないことです。1020行でＯＰＥＮしたバッファがＣＬＯＳＥされていれば、１番バッファを使ってもかまいませんし、３番バッファがあいていれば、このバッファでもかまいません。要は、同時に２つのファイルで同一バッファを使うことができないということです。

1020行と2020行でＯＰＥＮするファイル名は同じですが、それぞれフィールド設定が異なるので、同じバッファを同時に使うことができないのです。

2030行から2050行のＦＯＲ～ＮＥＸＴループで、性別を見分けるチェッカー用変数のフィールドを設定しています。

これはもうわかりますね。256回ループさせて、ＳＥＸ＄(１)からＳＥＸ＄(256)までの配列にそれぞれ１バイトずつフィールドを割り当てています。

この256回のループの過程で、ダミー用変数ＤＵＭＭＹ２＄のフィールドは０から始まり255まで変化し、最終的には255が残ります。

この部分のフィールド設定では、256バイトまるまる使っています。未使用バイト０ですから、すごく効率よく使うことができるわけです。

以上、ファイルのＯＰＥＮとフィールド設定、ご理解いただけましたか？　わかりやすいようにと気を配ったつもりですが、もし行き届

かない点がありましたら、もう一度最初から読みなおしてください。あるいは前著を参考にされるのもよいかと思います。

データ入力部分のプログラム

この部分は、みなさんはすっかりマスターされていると思います。次のようにキーインすればよいですね。

```
190 '
200 LOCATE 25,4 : PRINT "ﾃﾞｰﾀ ｦ ﾆｭｳﾘｮｸ ｼﾏｽ"
210 FOR I=1 TO 3000 : NEXT I
220 '
230 *LOOP
240 CLS
250 LOCATE 15,4 : INPUT "ｺｰﾄﾞ(END=999)";CD
260 IF CD=999 THEN GOTO *FCLOSE
270 LOCATE 15,5 : INPUT "ｾｯｸｽ<m/f>····";A1$
280 LOCATE 15,6 : INPUT "ﾅﾏｴ··········";A2$
290 LOCATE 15,7 : INPUT "ﾕｳﾋﾞﾝﾊﾞﾝｺﾞｳ··";A3$
300 LOCATE 15,8 : INPUT "ｼﾞｭｳｼｮ·······";A4$
310 LOCATE 15,9 : INPUT "ﾊﾞﾝﾁ·········";A5$
320 LOCATE 20,11 : PRINT "SURE < y/n > ?"
330 Y$=INKEY$ : IF Y$="" THEN GOTO 330
340 IF Y$="N" OR Y$="n" THEN GOTO *LOOP
350 IF Y$="Y" OR Y$="y" THEN GOTO *DAT.WRITE
360 BEEP : BEEP : GOTO 320
```

250行でコードを入力し、もし「999」がキーインされたら、現在ＯＰＥＮしているファイルをすべてＣＬＯＳＥして、このプログラムの処理を終りにします。この判定は、260行のＩＦ文で行なっています。

270行から310行までは登録する人のデータ入力です。

330行はＩＮＫＥＹ＄文によるキー入力待ちですね。何かキー入力があるまで、この330行から抜け出すことができせん。

320行の確認メッセージに対して340行で「Ｎ」または「ｎ」が入力された場合には、もう一度入力しなおさなければなりません。修正のステートメントを追加して、すべてのデータをキーインしないですむようにもできますが、ここでは省略します。みなさんが、あとでチャレンジしてみてください。

ＯＫであれば350行でデータをディスクに書き込むルーチンに飛ばします。360行は、とんでもないキーを押してしまったときの処理ですね。

ディスクへのデータ書き込み

さあ、データ入力部分ができました。次は、入力データの書き込みです。

書き込みは、コードはシーケンシャルファイルですからＷＲＩＴＥ文で、また性別、氏名、郵便番号、住所、番地はランダムファイルですからＬＳＥＴ文を使えばよかったですね。

ただ、前著と違うところは、ランダムファイルにおいて、性別を示すチェッカーは第１セクタに、氏名その他は第２セクタ以降にというように、それぞれ異なるセクタに同時に書き込まなければならないという点です。

ですから、書き込みの前にまず、書き込むべき位置を決めておかなければなりません。

では、プログラムをキーインする前に、この問題を解決しておきましょう。

最初に、性別を識別するためのチェッカーですが、これは第１セクタに入力された順に書き込んでいけばよいわけですね。ですから、入力順を表わす変数をＯＲＤＥＲ１とでもして、１つずつカウントしていけばよいのです。

いいですか？　この変数をカウンタ代わりにしていますから、１番目の人から順に、１、２、３……とカウントされ、このときの値が変数ＯＲＤＥＲ１に入ります。

つまり、３人目の人のときは、この変数の中味は３になりますから、ＳＥＸ＄(３)にＡ１＄の中味をバッファ上でＬＳＥＴ（エルセット）すればよいのです。

性別チェッカーの書き込みはわかりましたね。

次は第２セクタ以降です。いいですか。

第２セクタ以降には、１セクタ内に３人分のデータが書き込まれます。そして、第２セクタの頭から、１番目の人、２番目の人、３番目の人のデータと順に書き込まれ、４番目の人からは第３セクタに移ります。これを図式化すると次のようになります。黒くて太い数字が１番目の人、２番目の人……を表しています。



私たちが欲していることは、上の「１　２　３」のようにそれぞれ異なる数字を、セクタを表す共通の数「２」に換算できることです。

では、❶の数字を３で割って、その商を求めるとどうなるでしょうか。次のようになりますね。

0 0 1　1 1 2　2 2 3

３つ１組の数列のうちで１つだけが仲間はずれになっています。たとえば、「0　0　1」の組の「1」のように……。

これを３つとも同じ商にすることができれば、その商に定数を加えてセクタ番号を算出することができます。

もう、どうすればよいかわかりますね。仲間はずれといっても１だけずれているわけですから、❶の数列からあらかじめ１だけ引いてずらしておき、それを３で割って商を求めればよいわけですね。

やってみましょう。

❶の数列からそれぞれ１を引くと

0 1 2　3 4 5　6 7 8

となります。

では、この数列を３で割って商を並べてみますと、

0 0 0　1 1 1　2 2 2

となりますね。どうですか？　きれいにそろったでしょう。

それではこの数列に定数「2」を加えてみましょう。

2 2 2　3 3 3　4 4 4

第２セクタ　第３セクタ　第４セクタ

定数を加算したら、それぞれの数字が、それぞれ書き込まれるべきセクタ番号になりました。やっと私たちの念願がかなったわけですね。ここまでくれば、あとは楽です。

❶の数列は、入力される順番ですから、変数ＯＲＤＥＲ１にしまわれています。ですから、この変数を使えば、書き込まれるセクタ番号は次のように表わすことができます。

```
SEC=(ORDER1-1)¥3+2
```

どうやら峠を越したようですが、もう１つ問題が残っています。それは、書き込まれるセクタはわかりましたが、そのセクタ内で何番目

の位置に書き込んだらよいか、ということです。

これも、いまと同じように、25ページの❶の数列をもとに換算する式を作ってみましょう。

❶の数列で表わされる人たちのデータは、次のように、各セクタ内の緑色の太い数字で示された位置に書き込めばよいのです。

第2セクタ　第3セクタ　第4セクタ

1 2 3 4 5 6 7 8 9 ―― ❷

↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓

1 2 3 1 2 3 1 2 3 ―― ❸

今度は、❷の数列を3で割って、その余りを求めてみましょう。前は商でしたが、今度は剰余ですよ。

1 2 0 1 2 0 1 2 0

上の❸の数列の「3」に相当する部分がすべて「0」になっています。この「0」を「3」にすればよいのですが、3で割り算をしているのですから、余りが3になることはありませんね。

しかたないですから、余りが左から右へ昇順に並ぶように調整します。❷の数列から1を引いて、それを3で割り余りを求めてみてください。次のようになりますね。

0 1 2 0 1 2 0 1 2

どうですか？　昇順に並びましたね。そうしたら、この剰余にそれぞれ1を加えれば、❸で示す数列が得られますから、この方法によって、セクタ内で書き込まれる位置が決定できるわけです。

書き込まれる位置を示す変数をＯＲＤＥＲ２とすれば、次のようにして換算することができます。

```
ORDER2=(ORDER1-1) MOD 3+1
```

さあ、これでプログラムの下準備ができました。

少々、まわりくどい説明でしたが、ご理解いただけましたか？

ランダムファイルを使ってセクタを有効に、かつ便利に活用しようと思うと、どうしても、いま説明した書き込みセクタの算出と、セクタ内で何番目に書き込んだらよいかの位置の算出はまぬがれません。もし、このような方法を使わないとすると、ＩＦ文ばかり多くなり、大変非効率的なプログラムになってしまいます。

それでは次に、書き込みの入口部分のプログラムを示します。

```
370 '
380 *DAT.WRITE
390 ORDER1=ORDER1+1 ←カウンタ
400 SEC=(ORDER1-1)¥3+2 ←書き込まれるセクタ番号
410 ORDER2=(ORDER1-1) MOD 3+1 ←セクタ内での書き込まれる位置
420 IF ORDER2=1 THEN GOTO 440
430 GET #1,SEC : GET #2,1
440 WRITE #3,CD ←コードの書き込み
```

行番号 390 から 410 までが、これまでにご説明した部分です。

説明は長かったのですが、まとめてみるとこんなに短かくなってしまうのですから、ちょっとがっかりしますね。

行番号 420 はわかりますか？

そうです。あるセクタの１番目のデータを書き込むときには、430 行でＧＥＴしようとすると、

```
Input past end in 430
```

のエラーになってしまいますから、430 行を回避しているのですね。

そして、２番目、３番目のデータの場合は 430 行を通過させて、その書き込むデータのセクタをＧＥＴします。

このプログラムは初回の書き込みのためのものですから、430 行はなくても正常に動作すると思いますが、データ追加などを考慮すると、追加データを、あるセクタの３番目くらいから書き込む場合もありますので、このようなときにはこのセクタをＧＥＴしておかないと、すでに書き込まれている１番目、２番目のデータが消えてしまいます。すでに書き込まれているデータがあるときには必ずそのセクタをＧＥＴしてからＰＵＴするということ、これはプログラミングの過程でつい忘れられがちですが、注意してください。

440 行でコードをシーケンシャルファイルに書き込み、次に示すプログラムで各データをランダムファイルに書き込みます。

```
450 LSET SEX$(ORDER1)=A1$   ┐
460 LSET NM$(ORDER2) =A2$   │
470 LSET PN$(ORDER2) =A3$   ├ バッファ上でデータをセット
480 LSET AD$(ORDER2) =A4$   │
490 LSET BA$(ORDER2) =A5$   ┘
500 PUT #1,SEC : PUT #2,1 ←ランダムファイルへの書き込み
510 GOTO *LOOP
```

450行から490行で、各データをバッファ上でＬＳＥＴしていますが、Ａ１＄は２番バッファで、Ａ２＄からＡ５＄は１番バッファで、それぞれフィールド設定された変数にセットしています。

そして、このバッファ上でセットされたデータを、500行のＰＵＴ文でディスクに書き込んでいます。

510行は、データ入力へ戻しています。

エンディング

これでメイン部分は終わりましたので、次はエンド処理です。

まず、250行でコードに「999」がキーインされた場合、現在ＯＰＥＮしているファイルをＣＬＯＳＥしなければなりませんから、次のステートメントをキーインします。

```
520 '
530 *FCLOSE
540 CLOSE #1,#2,#3
```

540行ではＣＬＯＳＥするバッファ番号をコンマで区切って連記してありますが、これを省略した場合には、現在ＯＰＥＮしているバッファをすべてＣＬＯＳＥすることになります。すべての処理がこれで終りというのなら省略してもよいのですが、プログラム処理の途中ではこまめに入れておいたほうがよさそうです。というのは、そうしておけば、プログラムを見た場合に、バッファの使用状況がよくわかるからです。

次はおなじみのフィニッシュ部分です。

これは説明しなくともわかると思います。600行でメニュープログラムをＲＵＮさせていますが、これはあとで作ります。

```
550 '
560 CLS
570 LOCATE 25,4 : PRINT "ﾎｶ ﾉ ｼｺﾞﾄ ｦ ｼﾏｽｶ ?"
580 LOCATE 30,5 : PRINT "< y/n >"
590 Y$=INKEY$ : IF Y$="" THEN GOTO 590
600 IF Y$="Y" OR Y$="y" THEN RUN "DMMENU.bas"
610 IF Y$="N" OR Y$="n" THEN GOTO *FINISH
620 BEEP : BEEP :GOTO 570
630 '
640 *FINISH
650 END
```

データ入力のプログラムリスト

これで、まだ足りない点もあると思いますが、とりあえず完成ということにして、この部分のプログラムリストを次に示します。

```
10 ' file name ==> DM01/1.bas
20 ' 1984.1.10
30 ' random data write
40 ' save "DM01/1.bas"
50 '
100 CONSOLE 0,25,0,0 : WIDTH 80,25
110 DEFINT A-Z
120 DIM NM$(3),PN$(3),AD$(3),BA$(3)
130 DIM SEX$(256)
140 '
150 CLS
160 GOSUB *OPNF01
170 GOSUB *OPNF02
180 OPEN "2:INDX01.dat" FOR OUTPUT AS #3
190 '
200 LOCATE 25,4 : PRINT "ﾃﾞｰﾀ ｦ ﾆｭｳﾘｮｸ ｼﾏｽ"
210 FOR I=1 TO 3000 : NEXT I
220 '
230 *LOOP
240 CLS
250 LOCATE 15,4 : INPUT "ｺｰﾄﾞ(END=999)";CD
260 IF CD=999 THEN GOTO *FCLOSE
270 LOCATE 15,5 : INPUT "ｾｯｸｽ<m/f>････";A1$
280 LOCATE 15,6 : INPUT "ﾅﾏｴ･･････････";A2$
290 LOCATE 15,7 : INPUT "ﾕｳﾋﾞﾝﾊﾞﾝｺﾞｳ･･";A3$
300 LOCATE 15,8 : INPUT "ｼﾞｭｳｼｮ･･･････";A4$
310 LOCATE 15,9 : INPUT "ﾊﾞﾝﾁ･････････";A5$
320 LOCATE 20,11 : PRINT "SURE < y/n > ?"
330 Y$=INKEY$ : IF Y$="" THEN GOTO 330
340 IF Y$="N" OR Y$="n" THEN GOTO *LOOP
350 IF Y$="Y" OR Y$="y" THEN GOTO *DAT.WRITE
360 BEEP : BEEP : GOTO 320
370 '
380 *DAT.WRITE
390 ORDER1=ORDER1+1
400 SEC=(ORDER1-1)¥3+2
410 ORDER2=(ORDER1-1) MOD 3+1
```

```
420 IF ORDER2=1 THEN GOTO 440
430 GET #1,SEC : GET #2,1
440 WRITE #3,CD
450 LSET SEX$(ORDER1)=A1$
460 LSET NM$(ORDER2) =A2$
470 LSET PN$(ORDER2) =A3$
480 LSET AD$(ORDER2) =A4$
490 LSET BA$(ORDER2) =A5$
500 PUT #1,SEC : PUT #2,1
510 GOTO *LOOP
520 '
530 *FCLOSE
540 CLOSE #1,#2,#3
550 '
560 CLS
570 LOCATE 25,4 : PRINT "ﾎｶ ﾉ ｼｺﾞﾄ ｦ ｼﾏｽｶ ?"
580 LOCATE 30,5 : PRINT "< y/n >"
590 Y$=INKEY$ : IF Y$="" THEN GOTO 590
600 IF Y$="Y" OR Y$="y" THEN RUN "DMMENU.bas"
610 IF Y$="N" OR Y$="n" THEN GOTO *FINISH
620 BEEP : BEEP :GOTO 570
630 '
640 *FINISH
650 END
1000 '
1010 *OPNF01
1020 OPEN "2:MAIL01.dat" AS #1
1030 FOR I=1 TO 3
1040   FIELD #1,85*(I-1) AS DUMMY1$,20 A
S NM$(I),6 AS PN$(I),39 AS AD$(I),20 AS
BA$(I)
1050 NEXT I
1060 RETURN
2000 '
2010 *OPNF02
2020 OPEN "2:MAIL01.dat" AS #2
2030 FOR I=1 TO 256
2040   FIELD #2,(I-1) AS DUMMY2$,1 AS SEX$(I)
2050 NEXT I
2060 RETURN
```

1・3　プログラムの実行

セクタの扱いを変えた初めてのプログラムです。さっそく実行してみたいと思いますが、その前に、まずプログラムをディスクにセーブしておきましょう。

```
save "DM01/1.bas"
Ok
■
```

セーブできましたら、いよいよ実行です。

入力するデータですが、次の５人分とします。

コード	性別	氏　名	郵便番号	住　所	番　地
101	m	AAAAAAA	111	aaaaaaaaaa	1-1-1
201	m	BBBBBBB	222	bbbbbbbbbb	2-2-2
301	f	CCCCCCC	333	cccccccccc	3-3-3
401	f	DDDDDDD	444	dddddddddd	4-4-4
501	m	EEEEEEE	555	eeeeeeeeee	5-5-5

データといっても、氏名、住所がアルファベットでちょっと実感がわきませんが、このプログラムが確実に動作するかを確認するだけですから、入力しやすいほうがよいでしょう。

さあ、それでは f・5 キーを押してプログラムの実行です。

おや？　エラーメッセージがでてきました。

```
Disk offline in 1020
Ok
■
```

うっかりしていましたね。**データファイル用のディスクをドライブ２にまだセットしてありませんでした。**そのために、1020行でドライブ２側のランダムファイルをＯＰＥＮしようとしたところ、ディスクがつながっていない、というエラーとなったのです。

このような手違いはよくありますので、ディスクをセットしたかどうかの確認をうながすステートメントをあとで作ることにしましょう。

それではディスクをドライブ２にセットしてもう一度ＲＵＮです。

実行すると、アクセス表示用ＬＥＤが点灯し、ファイルをＯＰＥＮします。次に「データ　ヲ　ニュウリョク　シマス」という見出しが表示され、データ入力に移ります。

入力方法はもうおわかりと思います。前ページのデータをメッセージに従ってキーインしていってください。

たとえば、１番目のデータについては次のような画面になります。

```
ｺｰﾄﾞ(END=999)? 101
ｾｯｸｽ<m/f>····? m
ﾅﾏｴ··········? AAAAAAA
ﾕｳﾋﾞﾝﾊﾞﾝｺﾞｳ··? 111
ｼﾞｭｳｼｮ·······? aaaaaaaaaa
ﾊﾞﾝﾁ·········? 1-1-1

      SURE < y/n > ?
```

間違いがなければ、大文字でも小文字でもよいですから「Y」をキーインします。すると、カチッと音がしてランダムファイルにデータが書き込まれ、ふたたびデータ入力画面になります。

５番目のデータまで入力できましたら、次のコードの入力メッセージに対して「999」をキーインします。すると、データ入力処理を終り、「ホカ　ノ　シゴト　ヲ　シマスカ　？」というメッセージが表示されますから、「N」を入力してください。

なお、何かのつごうでデータ入力を最初からやりなおしたいときは、STOP キーでプログラムの実行を止め、ダイレクトモードでファイルをＣＬＯＳＥし、ランダムファイル、シーケンシャルファイルをＫＩＬＬコマンドで削除してから再度ＲＵＮしてください。

```
close
Ok
kill "2:INDX01.dat"
Ok
kill "2:MAIL01.dat"
Ok
█
```

1・4　書き込みデータの確認

前節でデータの入力が終りましたが、正確に書き込まれているかどうか、ちょっと不安ですね。

この節では、一時しのぎの方法で確認してみることにします。

確認の手順

シーケンシャルファイルについては、まずダイレクトモードでファイルをＯＰＥＮし、ＦＯＲ～ＮＥＸＴループを使って５個のデータを読み出し、それをＰＲＩＮＴ文で画面に表示させます。

ランダムファイルについても手順は同じで、ファイルをＯＰＥＮして必要なセクタをＧＥＴすればよいのですが、フィールド設定などがあり、ダイレクトモードでキーインするにはちょっとシンドイので、私たちが作ったプログラムの一部を流用することにします。

シーケンシャルファイルのデータ確認

まず、ダイレクトモードで次のようにキーインし、シーケンシャルファイルをＩＮＰＵＴ（読み出し）モードでＯＰＥＮします。現在入力してあるプログラムは消さずにそのままにしておいてください。

```
open "2:INDX01.dat" for input as #1 ⏎
```

（input ── データ読み出しモード）

カチッと音がして、インデックス用のシーケンシャルファイルがオープンされました。

「Ok」が表示されたら、ダイレクトモードで次のようにキーインします。

```
for i=1 to 5:input #1,cd:? cd;"  ";:next
```

（input #1,cd ── データ読み出し／? cd;" " ── データを表示／; ── 改行せずに表示させるために入れる）

どうですか？　リターンキーを押すやいなやデータが表示されましたね。そしてそれは、まぎれもなく私たちが入力したコードです。

これで、コードは無事読み込めていることがわかりましたので、ファイルをクローズしましょう。次のようにキーインします。

```
close #1 ⏎
```

それではダイレクトモードでキーインした結果を画面表示で示しておきます。同じになりましたか？

```
open "2:INDX01.dat" for input as #1
Ok
for i=1 to 5:input #1,cd:? cd;"  ";:next
 101    201    301    401    501
Ok
close #1
Ok
█
```

ランダムファイルでデータ確認

次はランダムファイルの場合ですが、今度は、プログラムモードとダイレクトモードの混合で行ないます。

最初に、現在入力してあるプログラムの不要部分を削除します（プログラムを消してしまった人は、「ＤＭ０１/１．bas」をロードしてください）。次のようにキーインしてください。

```
delete 10-650 ⏎
```

キーインできましたら、ＬＩＳＴコマンドでプログラムリストを画面に表示させてみてください。

行番号1000行以降のファイルオープンとフィールド設定のステートメントだけが残りましたね。

ランダムファイルではフィールド設定がありますから、ダイレクトモードでキーインするのはめんどうなので、ステートメントがせっかくそろっているのですから、これを流用します。

ただし、この部分はサブルーチンになっていますから、1060、2060行のＲＥＴＵＲＮ文は削除しておきます。その方法は、削除したい行番号を入力してリターンでしたね。

```
1060 ⏎
2060 ⏎
```

さあ、もう一度プログラムリストを表示させてみてください。

```
list
1000 '
1010 *OPNF01
1020 OPEN "2:MAIL01.dat" AS #1
1030 FOR I=1 TO 3
1040   FIELD #1,85*(I-1) AS DUMMY1$,20 A
S NM$(I),6 AS PN$(I),39 AS AD$(I),20 AS
BA$(I)
1050 NEXT I
2000 '
2010 *OPNF02
2020 OPEN "2:MAIL01.dat" AS #2
2030 FOR I=1 TO 256
2040   FIELD #2,(I-1) AS DUMMY2$,1 AS SEX$(I)
2050 NEXT I
Ok
■
```

リストをごらんください。1000行から、および2000行からのプログラムでうまくファイルをＯＰＥＮできそうですね。

f・5 キーを押して実行してみましょう。

おやッ、エラーメッセージがでてしまいました。

```
run
Subscript out of range in 2040
Ok
■
```

2040行で配列の添字（そえじ）不足ということですね。

上のリストをごらんください。2030行のＦＯＲ文でＩは256回ループすることになっています。したがって、配列も256の大きさで宣言しておかなければなりません。ダイレクトモードで、まず**ファイルをクローズし、次にＣＬＥＡＲ（注）してから配列を宣言します。**

〈注〉ＣＬＥＡＲ文で配列変数などを消去してから配列宣言をしないと、配列二重定義のエラーになります。

```
clear
Ok
dim sex$(256)
Ok
■
```

さあ、これでだいじょうぶだと思いますが、今度も前と同じように（f・5）キーでRUNさせると、同じ配列の添字不足のエラーになってしまいますよ。

なぜだかわかりますか？　そうですね、RUNコマンドを実行するとすべて初期状態からスタートすることになるのでした。ですから、せっかく宣言した配列も無効になってしまうのです。

しかたがないですから、GOTO文を使います。次のステートメントをキーインしてください。

```
go to 1000
```

カチッと音がしてランダムファイルがオープンされました。今度はエラーになりませんでした。

それでは最初に、第1セクタに書き込まれているはずの性別用のチェッカーを調べてみましょう。次のようにキーインし、第1セクタをGETします。

```
get #2,1
```

「Ok」が表示されましたら、続いて次のようにキーインしてください。

```
for i=1 to 5:? sex$(i);"   ";:next i
```

5人分のデータを入力したので、FOR～NEXTで5回ループさせ、バッファ上の配列変数の第1番目の要素から第5番目までの要素を見ようというわけです。

リターンキーを押すと、画面は次のようになりました。

```
go to 1000
Ok
get #2,1
Ok
for i=1 to 5:? sex$(i);"   ";:next i
m   m   f   f   m
Ok
■
```

どうですか？　うまく書き込まれていましたね。

私たちが入力した順に「m　m　f　f　m」と表示されました。

次は第２セクタ以降の様子を調べますが、全部をチェックしないで抜き取り検査式に１つだけ確かめてみましょう。

たとえば32ページのデータで、コード番号 401 の人について調べてみましょう。

この人のデータは４番目に入力しましたから、第３セクタの第１レコードになっているはずですね。ということは、第３セクタをＧＥＴして、フィールド設定された変数の添字１の中味を見ればよいことになります。

第２セクタ以降は１番バッファでファイルをＯＰＥＮしていますから、次のようにキーインします。バッファ番号を間違えないようにしてください。

```
get #1,3
```

「Ok」が表示されたら、続いて次のようにキーインしてください。

```
? nm$(1):? pn$(1):? ad$(1):? ba$(1)
```

リターンキーを押すと画面は次のようになりました。

```
get #1,3
Ok
? nm$(1):? pn$(1):? ad$(1):? ba$(1)
DDDDDDD
444
dddddddddd
4-4-4
Ok
█
```

４人目の人の項目

ランダムファイル第３セクタの第１レコードの内容を表示

ランダムファイルへの書き込みも異常ないようですね。

前に、バッファ１番においてはダミー変数ＤＵＭＭＹ１＄も活躍の権利をもっていると申し上げました。次のようにキーインしてみてください。この変数のフィールド幅は 170 バイトですので、第３セクタの第１、第２レコードが表示されます。

```
? dummy1$
```

表示されましたか？　これで前の話との連絡がついたと思います。

最後にファイルをクローズしておきましょう。

```
close
```

1·5 データの追加

私たちが1·2節で作ったプログラムは、初回のデータ書き込みしかできません。これではちょっと不便ですね。ディスクに記憶されているデータを修正したいとか、あるいはあらたに追加したいとか、いろいろ希望があるはずです。

そこでこの節では、30～31ページのプログラムリストに必要なステートメントを追加して、初回および追加時兼用のプログラムにしてみましょう。

それではまず、プログラムをロードしてください。

```
load "DM01/1.bas" ↵
```

ところであなたはプリンタをお持ちですか？　もし、プリンタの用意があるのでしたら、次のようにキーインしてプログラムリストをプリンタに出力し、印刷されたリストを見ながら読み進んでください。

```
llist ↵
```

プリンタをお持ちでない方は、しかたないですから、30～31ページのリストか、ディスプレイ上のリストを見ながら進んでください。

初回の書き込み？　それとも追加？

最初に、初めてのデータ入力か、それとも既存のファイルに追加するのか判定しなければなりませんね。

この方法は、前著をお読みになった方はおわかりと思いますが、ランダムファイルをOPENして、そのファイル長をLOF（レングス・オブ・ファイル）関数で調べ、もしその値が0ならばまだ一度も書き込みが行なわれていないわけですから、インデックス用のシーケンシャルファイルをOUTPUTモードで、その他の場合にはAPPENDモードでOPENします。

データを追加したいのにOUTPUTモードでOPENすると、ファイルポインタがシーケンシャルファイルの先頭にセットされ、以前に書き込んだデータが無効になってしまいます。

そして、データ追加の場合には、既存のファイルをINPUTモードでOPENし、書き込んであるデータ数を調べます。この数がわかれば、最初の追加データが何番目のものか決まりますから、これによって、ランダムファイルへの書き込み位置が換算されます。

この部分のステートメントは、ランダムファイルをＯＰＥＮしてからですから、170行のＧＯＳＵＢ文でサブルーチンに飛び、帰ってきたところへ入れます。

プログラムリストを次に示します。

```
171 IF LOF(2)=0 THEN GOTO 180 ELSE GET #2,1
172 GOSUB *READ.INDX : ORDER1=COUNT
173 OPEN "2:INDX01.dat" FOR APPEND AS #3
174 GOTO 200
```

行番号171で、もし２番のランダムファイル長が０であれば、行番号180のＯＵＴＰＵＴモードへ飛ばします。

追加の場合には172行へ処理が落ちてきますから、サブルーチンへジャンプさせ、すでに書き込みずみのデータ数をカウントします。

この数値は変数ＯＲＤＥＲ１に代入され、行番390でカウンタを１上げていますから、ランダムファイルには既存のデータの次から書き込まれることになります。

そして行番号173で、シーケンシャルファイルをＡＰＰＥＮＤモードでＯＰＥＮします。

それでは、行番号172の飛び先であるサブルーチンを作っておきましょう。プログラムは次のとおりです。

```
3000 '
3010 *READ.INDX
3020 OPEN "2:INDX01.dat" FOR INPUT AS #3
3030 WHILE NOT(EOF(3))
3040   COUNT=COUNT+1
3050   INPUT #3,DUMMY
3060 WEND
3070 CLOSE #3
3080 RETURN
```

3030行と3060行のＷＨＩＬＥ～ＷＥＮＤ文はわかりますね。

ＷＨＩＬＥのうしろの条件が満たされている間は、ＷＥＮＤとの間のステートメントを実行します。この場合、シーケンシャルファイルの末尾のデータを読み出すまでループします。

この部分は、データ数を数えるだけですから、3050行でデータを読むときも変数はダミーでいいですね。

そして、ＲＥＴＵＲＮ文でもとに戻るときに、ファイルをクローズ

するのを忘れないでください。

ディスクのセットはOK？

どこでしたか、「Disk offline」のエラーがでてきてあわてたことがありましたね。このようなミスが起こらないように、確認のメッセージを入れておきましょう。

これは当然、ファイルをＯＰＥＮする前に入れなければなりませんね。140行と150行の間に入れることにします。

```
141 CLS
142 LOCATE 25,4 : PRINT "ﾄﾞﾗｲﾌﾞ2ﾍ ﾃﾞｨｽｸｦﾄﾞｳｿﾞ"
143 LOCATE 25,6 : PRINT "    Sure <y/n> ?"
144 Y$=INKEY$ : IF Y$="" THEN GOTO 144
145 IF Y$="Y" OR Y$="y" THEN GOTO 150
                        ELSE GOTO 142
146 '
```

これは、大文字か小文字の「Ｙ」が入力されたときのみ、この部分から抜け出せるようになっています。

あなたが、データファイル用ディスクをドライブ２にセットするまで、コンピュータは待っていてくれるのです。

これ以上データを書き込めない！

このプログラムでは、性別を見分けるチェッカー用として第１セクタだけを使っていますので、書き込むことができるデータ数は256まででです。

したがって、行番号390のカウンタ用変数ＯＲＤＥＲ１の内容が、256を越えてしまうとまずいわけです。ですから、行番号を395として制限をつけておくことにします。

次のようにキーインしてください。

```
395 IF ORDER1>256 THEN GOTO *ACCIDENT
```

万一このようなことが起こったら、プログラムのうしろのほうへ飛ばして、一時的に処理をしておきましょう。

このような事故を防ぐためには、できるだけ長期的な立場に立って考え、第２セクタ、第３セクタをもチェッカー用に確保しておくとよいと思います。

次に、395行からのジャンプ先のステートメントを示します。

```
4000 '
4010 *ACCIDENT
4020 CLOSE #1,#2,#3
4030 CLS : BEEP : BEEP
4040 LOCATE 25,4 : PRINT "デ゛ータ カキコミ デ゛キマセン!"
4050 FOR I=1 TO 3000 : NEXT I
4060 GOTO 560
```

4020行で、とりあえず現在ＯＰＥＮしているファイルをＣＬＯＳＥしています。そしてメッセージを表示し、560行へジャンプさせています。

プログラムの実行

<注>前節と同じファイル名でセーブすると前のプログラムは消えてしまいますが、この節でのプログラムが正常に走れば、前のものは使う機会を失いますのでかまわないでしょう。

さあ、これでだいたいプログラムの修正がすみましたので、これを実行して動作テストをしてみましょう。

その前に、**このプログラムを一応セーブしておいてください。**ファイル名は前と同じでかまいません。<注>

セーブできましたら、前と同じように、次の２人分のデータを追加してみましょう。

601	f	FFFFFFF	666	ffffffffff	6-6-6
701	m	GGGGGGG	777	gggggggggg	7-7-7

これは、６人目と７人目のデータになりますね。入力方法はもうわかりますね。ＲＵＮさせてメッセージに従ってキーインしてください。

入力が終りましたら、35ページで学んだ方法でデータの確認をしてみます。まず、次のように不要な行をデリートします。

```
delete 10-650
Ok
delete 3000-4060
Ok
1060
2060
█
```

そうしたら、続いて次のようにダイレクトモードでキーインし、７つ目のデータが書き込まれる第４セクタをＧＥＴしてみます。

```
clear ←――――――変数をクリア
Ok
dim sex$(256) ←――――配列宣言
Ok
go to 1000 ←――――1000行から実行
Ok
get #1,4 ←――――――第４セクタをＧＥＴ
Ok
for i=1 to 3:? nm$(i);pn$(i):next
GGGGGGG              777 ←――――７人目のデータ
EEEEEEE              555 ┐
FFFFFFF              666 ┘―――５人目、６人目のデータが書き込まれている
Ok
close ←――――――――ファイルをＣＬＯＳＥ
Ok
■
```

上の画面をごらんください。

７人目のデータまで入力したのですから、第４セクタは１番目のレコードである７人目のデータ以外は書き込まれてはいけないわけです。しかし画面では、５人目、６人目のデータが第２レコード、第３レコードとして表示されています。〈注〉

〈注〉ランダムファイル上では８人目、９人目の書き込み位置にデータが書かれてしまったことになりますが、インデックスには登録されていませんから別に支障はないのですが、気分的にスッキリしませんね。

このようになってしまった原因を知るのに、プログラムリストの必要な個所をもう一度示します。

```
420 IF ORDER2=1 THEN GOTO 440
430 GET #1,SEC : GET #2,1
440 WRITE #3,CD
450 LSET SEX$(ORDER1)=A1$
460 LSET NM$(ORDER2) =A2$
470 LSET PN$(ORDER2) =A3$
480 LSET AD$(ORDER2) =A4$
490 LSET BA$(ORDER2) =A5$
500 PUT #1,SEC : PUT #2,1
```

420行で、第１レコードの書き込みであったら、430行のＧＥＴ文を通さないで440行へ飛ばしています。そして、450行以下でデータをバッファ上でＬＳＥＴしています。

しかし、ここで注意してください。このとき、第1レコードのデータは確実にセットされますが、第2レコード、第3レコードのバッファ上の変数には何が入っているでしょうか？

7人目の1つ前、6人目のデータをセットするとき、430行で第3セクタをGETしていますから、このときの状態が、7人目のデータ書き込みのときにもそのまま持ち込まれています。

つまり、7人目のデータを書き込むときも、第2レコード、第3レコード用のバッファ上の変数に、第3セクタの第2レコード、第3レコードのデータが保存されていたのです。

ですから、第1レコードの書き込みの場合には、LSETする前に、第2レコード、第3レコード用の変数をバッファ上でクリアしておかなければならないのです。

さあ、それでは次のステートメントを追加しましょう。

```
442 IF ORDER2<>1 THEN GOTO 450
443 FOR I=2 TO 3
444 LSET NM$(I)=""
445 LSET PN$(I)=""
446 LSET AD$(I)=""
447 LSET BA$(I)=""
448 NEXT I
```

この場合、ループ変数Iは「2」から始まりますから注意してください。また、444行から447行までのダブルクォーテーションの間には何も文字列が入っていません。つまり、ヌルストリングです。

これでうまく動作すると思います。

次の3名のデータを追加してみましょう。

801	m	HHHHHHH	888	hhhhhhhhhh	8-8-8
901	f	IIIIIII	999	iiiiiiiiii	9-9-9
001	f	JJJJJJJ	000	jjjjjjjjjj	0-0-0

●8人目、9人目のデータは、すでにデータが書き込まれている第4セクタの第2、第3レコードの位置に重ねて書き込まれ、前のデータは消失します。

データ入力が終わりましたら、もう一度、42〜43ページの方法で、今度は第5セクタをGETして調べてみてください。

最後にリナンバリングをして、このプログラムにピリオドを打ちましょう。リナンバリングしてセーブする前に次ページの注意書きをお読みください。

```
renum 100,100
renum 1000,840
renum 2000,1070
renum 3000,2070
renum 4000,3090
```

〈注意〉 リナンバリングしたプログラムだけをセーブしておくと、あとでもう一度この本を勉強する場合に、解説部分で行番号のくいちがいがでてきますので、できたら違うファイル名でもセーブしておくと便利です。つまり、まったく同じ処理をするプログラムが違うファイル名で同一ディスクに２つ存在することになりますが、容量は十分ありますから心配いりません。

```
10 ' file name ==> DM01/1.bas
20 ' 1984.1.10
30 ' random data write
40 ' save "DM01/1.bas"
50 '
100 CONSOLE 0,25,0,0 : WIDTH 80,25
110 DEFINT A-Z
120 DIM NM$(3),PN$(3),AD$(3),BA$(3)
130 DIM SEX$(256)
140 '
150 CLS
160 LOCATE 25,4 : PRINT "ﾄﾞﾗｲﾌﾞ2ﾍ ﾃﾞｨｽｸｦﾄﾞｳｿﾞ"
170 LOCATE 25,6 : PRINT "     Sure <y/n> ?"
180 Y$=INKEY$ : IF Y$="" THEN GOTO 180
190 IF Y$="Y" OR Y$="y" THEN GOTO 210
                        ELSE GOTO 160
200 '
210 CLS
220 GOSUB *OPNF01
230 GOSUB *OPNF02
240 IF LOF(2)=0 THEN GOTO 280 ELSE GET #2,1
250 GOSUB *READ.INDX : ORDER1=COUNT
260 OPEN "2:INDX01.dat" FOR APPEND AS #3
270 GOTO 300
280 OPEN "2:INDX01.dat" FOR OUTPUT AS #3
290 '
300 LOCATE 25,4 : PRINT "ﾃﾞｰﾀ ｦ ﾆｭｳﾘｮｸ ｼﾏｽ"
310 FOR I=1 TO 3000 : NEXT I
320 '
330 *LOOP
```

```
340 CLS
350 LOCATE 15,4 : INPUT "ｺｰﾄﾞ(END=999)";CD
360 IF CD=999 THEN GOTO *FCLOSE
370 LOCATE 15,5 : INPUT "ｾｯｸｽ<m/f>････";A1$
380 LOCATE 15,6 : INPUT "ﾅﾏｴ･･････････";A2$
390 LOCATE 15,7 : INPUT "ﾕｳﾋﾞﾝﾊﾞﾝｺﾞｳ･･";A3$
400 LOCATE 15,8 : INPUT "ｼﾞｭｳｼｮ･･･････";A4$
410 LOCATE 15,9 : INPUT "ﾊﾞﾝﾁ･････････";A5$
420 LOCATE 20,11 : PRINT "SURE < y/n > ?"
430 Y$=INKEY$ : IF Y$="" THEN GOTO 430
440 IF Y$="N" OR Y$="n" THEN GOTO *LOOP
450 IF Y$="Y" OR Y$="y" THEN GOTO *DAT.WRITE
460 BEEP : BEEP : GOTO 420
470 '
480 *DAT.WRITE
490 ORDER1=ORDER1+1
500 IF ORDER1>256 THEN GOTO *ACCIDENT
510 SEC=(ORDER1-1)¥3+2
520 ORDER2=(ORDER1-1) MOD 3+1
530 IF ORDER2=1 THEN GOTO 550
540 GET #1,SEC : GET #2,1
550 WRITE #3,CD
560 IF ORDER2<>1 THEN GOTO 630
570 FOR I=2 TO 3
580 LSET NM$(I)=""
590 LSET PN$(I)=""
600 LSET AD$(I)=""
610 LSET BA$(I)=""
620 NEXT I
630 LSET SEX$(ORDER1)=A1$
640 LSET NM$(ORDER2) =A2$
650 LSET PN$(ORDER2) =A3$
660 LSET AD$(ORDER2) =A4$
670 LSET BA$(ORDER2) =A5$
680 PUT #1,SEC : PUT #2,1
690 GOTO *LOOP
700 '
710 *FCLOSE
720 CLOSE #1,#2,#3
730 '
740 CLS
750 LOCATE 25,4 : PRINT "ﾎｶ ﾉ ｼｺﾞﾄ ｦ ｼﾏｽｶ ?"
```

```
760 LOCATE 30,5 : PRINT "< y/n >"
770 Y$=INKEY$ : IF Y$="" THEN GOTO 770
780 IF Y$="Y" OR Y$="y" THEN RUN "DMMENU.bas"
790 IF Y$="N" OR Y$="n" THEN GOTO *FINISH
800 BEEP : BEEP :GOTO 750
810 '
820 *FINISH
830 END
1000 '
1010 *OPNF01
1020 OPEN "2:MAIL01.dat" AS #1
1030 FOR I=1 TO 3
1040   FIELD #1,85*(I-1) AS DUMMY1$,20 A
S NM$(I),6 AS PN$(I),39 AS AD$(I),20 AS
BA$(I)
1050 NEXT I
1060 RETURN
2000 '
2010 *OPNF02
2020 OPEN "2:MAIL01.dat" AS #2
2030 FOR I=1 TO 256
2040   FIELD #2,(I-1) AS DUMMY2$,1 AS SEX$(I)
2050 NEXT I
2060 RETURN
3000 '
3010 *READ.INDX
3020 OPEN "2:INDX01.dat" FOR INPUT AS #3
3030 WHILE NOT(EOF(3))
3040   COUNT=COUNT+1
3050   INPUT #3,DUMMY
3060 WEND
3070 CLOSE #3
3080 RETURN
4000 '
4010 *ACCIDENT
4020 CLOSE #1,#2,#3
4030 CLS : BEEP : BEEP
4040 LOCATE 25,4 : PRINT "ﾃﾞｰﾀ ｶｷｺﾐ ﾃﾞｷﾏｾﾝ!"
4050 FOR I=1 TO 3000 : NEXT I
4060 GOTO 740
```

1·6　データの修正

データインプットのプログラムが完成しましたので、次は、すでにディスクに書き込んであるデータを修正したい場合を考えてみましょう。

前著でも取り上げましたので、どうすればよいかすでに目星がついている方もいらっしゃるかもしれません。

しかし、今回は、ランダムファイルの異なるセクタに同時に書き込むという点で前回とは多少違いますし、また同じようなことを何回もくりかえすことが、プログラミングマスターへの着実な道だと思いますので、ぜひおつきあいしてください。

正確なデータが得られないときに、ダイレクトモードでどうチェックすればよいかなども取り上げるつもりです。

データ追加のプログラムを流用

データ修正の基本的な手順は、まず第1にインデックス用のシーケンシャルファイルからデータを読み込み、それを配列にしまいます。次に、修正したい人のコードを入力し、そのコードによってランダムファイルからデータを取り出してきて、画面で確認しながら新しいデータを入力していきます。

そして、修正データをキーインし終ったら、ランダムファイルの該当セクタに書き込みます。

以上が、データ修正のおおまかな流れです。

ここで、ちょっと考えてください。

データ修正の場合も、データ追加のときと同じファイルを使うわけですね。そして当然、ファイルをＯＰＥＮしなければなりません。

ということは、ファイルをＯＰＥＮするルーチンとか、初期設定の部分は、データ追加のプログラムがそのまま使えそうです。

45ページから47ページまでのプログラムリストを見てください。

❶　行番号10—50は作りなおしましょう。

❷　行番号100 — 230は初期設定ですから、そのまま使えます。

❸　シーケンシャルファイルにデータは書き込みませんから、行番号240および行番号260 — 280は不要です。

❹　300行のメッセージだけ修正すれば、行番号290 — 360は生かせます。

❺　行番号 370 から 620 までは修正ルーチンに変えなければなりませんので不要です。

❻　行番号 630 から3080まではそのまま使えます。 630 行からのＬＳＥＴ文も生きるように、変数のくふうをしてみましょう。

❼　行番4000からのサブルーチンは不要です。

どうですか？　だいぶ流用できる部分があるでしょう。

さあ、それではプログラミングにとりかかりましょうか。

まず、前のプログラムをロードしてください。

```
load "DM01/1.bas" ↵
```

ロードできましたら、次のようにダイレクトモードで、いま述べた不要な行を削除します。

```
delete 10-50
Ok
240
delete 260-280
Ok
delete 370-620
Ok
delete 4000-4060
Ok
█
```

そして、画面をクリアして、 200 行から 300 行くらいまでをＬＩＳＴコマンドで表示させてみてください。

```
list 200-
200 '
210 CLS
220 GOSUB *OPNF01
230 GOSUB *OPNF02
250 GOSUB *READ.INDX : ORDER1=COUNT
290 '
300 LOCATE 25,4 : PRINT "ﾃﾞｰﾀ ｦ ﾆｭｳﾘｮｸ ｼﾏｽ"
310 FOR I=1 TO 3000 : NEXT I
^C
Break    ――――ストップキーを押してブレークがかかった
Ok
█
```

●ＬＩＳＴコマンドで画面上に現れるリストを一時的にストップさせたいときは、コントロールキーを小指で押さえながら人差指でＳキーを押します。ここでリストコマンドを止めてしまいたかったらストップキーを、続行したければもう一度Ｓキーを押します。

前ページのような画面になりましたか？

まず、300行のメッセージをなおしておきましょう。次の緑色の部分のようにスクリーンエディタで修正してください。

```
300 LOCATE 25,4 : PRINT "ﾃﾞｰﾀ ﾉ ｼｭｳｾｲ ｦ ｼﾏｽ"
```

220行、230行のランダムファイルのＯＰＥＮはこのままでよいですね。

問題は、250行のシーケンシャルファイルを読み出すサブルーチンへジャンプする個所ですが、このデータ修正プログラムではコードを配列に読み込まなければなりませんので、コード用の配列宣言が必要です。しかし、これにはずいぶん便利な方法がありましたね。前著で説明しました。みなさん覚えていますか？

まず一度サブルーチンに飛ばしてデータのカラ読みをし、あわせてデータ数をカウントしてしまいます。そしてこのカウントされた数で配列を宣言します。

そして、一度サブルーチンへジャンプして帰ってきたら、フラグを立てておき、再度サブルーチンへジャンプしたときに、このフラグを配列にコードを読み込むかどうかの判定用に使います。

行番号250を次のように変更します。

```
250 GOSUB *READ.INDX : FLAG=1
```

次に、このサブルーチンを完成させてしまいます。完成させるといっても、ほとんど前のステートメントが使えるので簡単です。

プログラムリストを次に示しますが、緑色の2行を追加するだけでよいのです。

```
3000 '
3010 *READ.INDX
3020 OPEN "2:INDX01.dat" FOR INPUT AS #3
3030 WHILE NOT(EOF(3))
3040   COUNT=COUNT+1
3045   IF FLAG<>1 THEN GOTO 3050
3046   INPUT #3,CODE(COUNT)  ←—— コードを配列にしまう
3050   INPUT #3,DUMMY  ←—— データの読み飛ばし
3060 WEND
3070 CLOSE #3
3080 RETURN
```

第１回目のジャンプから戻ってきたら、カウンタ用変数に格納されている数だけコード用配列の宣言をします。

配列宣言をしたらカウンタをクリアして、もう一度サブルーチンへジャンプです。

カウンタのリセットを忘れると、プログラム完成後ＲＵＮさせた場合に、２回目のジャンプで、

```
Subscript out of range in 3046
```

のエラーになってしまいます。

次のプログラムを追加します。

```
260 DIM CODE(COUNT) : COUNT=0
270 GOSUB *READ.INDX
```

データサーチ

270行のサブルーチンへのジャンプで、登録してある人たちのコード番号を配列に読み込むことができましたので、これで修正作業への下準備ができたことになります。

次は、修正ルーチンを作るわけですが、これを挿入する付近のリストをＬＩＳＴコマンドで画面に表示させてみましょう。

```
list 330-
330 *LOOP
340 CLS
350 LOCATE 15,4 : INPUT "ｺｰﾄﾞ(END=999)";CD
360 IF CD=999 THEN GOTO *FCLOSE
630 LSET SEX$(ORDER1)=A1$
640 LSET NM$(ORDER2) =A2$
^C
Break
Ok
■
```

この間に修正ルーチンを挿入します

350行で、修正すべきデータのコードが入力されたら、360行で修正作業が終わりかどうかのチェックをし、終りでなければ、データサーチのサブルーチンに飛ばし、所定の人のデータを探し出します。

ここでは最初に、このサブルーチンを作ってしまいましょう。

ある配列にしまわれているコード番号の中から、キーインしたデータを探し出すにはどうしたらよいかわかりますか？

これには大変便利なSEARCH関数がありますので、これを使ってみましょう。

まず、次のようにダイレクトモードでキーインしてください。

```
for i=1 to 5:c%(i)=i:next i   ←整数配列C%に1から5までの数字を格納
Ok
? search(c%,3)   ←3という数値が配列内にあるかサーチ
 3   ←3番目にありました
Ok
■
```

●SEARCH関数は、1次元の整数型配列にのみ有効です。カッコの中の数式の並びは、上の例では配列名、サーチする整数となっていますが、さらにこのうしろに開始ナンバー、ステップ値などを指定することもできます。

この画面をごらんいただければわかると思いますが、C%(1)〜C%(5)までに1〜5の整数を代入し、SEARCH関数で3という整数がC%という配列内にあるかどうかサーチしています。

サーチののち、画面に表示された「3」という値は、配列の要素番号を示していますから、C%(3)の中にしまわれていることが確認できるわけです。反対にもし該当データがない場合には、「−1」が返されます。

さあ、それではSEARCH関数を使ってサブルーチンを完成させましょう。次のようにキーインしてください。

```
4000 '
4010 *DAT.SEARCH
4020 FOUND=SEARCH(CODE,CD)
4030 IF FOUND<>-1 THEN RETURN          ←見つかればリターン
4040 LOCATE 20,6 : BEEP : BEEP         ┐
4050 PRINT "ﾄｳﾛｸ ｼﾃ ｱﾘﾏｾﾝ!!"            ├見つからないときの処理
4060 FOR I=1 TO 3000 : NEXT I          ┘
4070 RETURN
```

プログラムは簡単ですからわかりますね。サーチする配列名はＣＯＤＥ、そしてこの中からＣＤを見つけます。配列名のうしろに%をつけないのは、プログラムヘッドでＤＥＦＩＮＴ文によって変数の整数宣言がしてあるからですね。

見つからないときには、ブザー音を鳴らして、注意をうながすメッセージを表示させましょう。

さて次は、このサブルーチンから帰ってからの処理です。

```
370 GOSUB *DAT.SEARCH
380 IF FOUND=-1 THEN GOTO *LOOP   ←見つからなければ
                                     コード再入力へ
390 ORDER1=FOUND   ←要素番号を代入
400 SEC=(ORDER1-1)¥3+2           ┐
410 ORDER2=(ORDER1-1) MOD 3+1    ┘セクタ、レコード番号の計算
420 GET #1,SEC : GET #2,1
```

もし、**該当コードが見つからなければ、**しかたないですから、**もう一度コード入力ルーチンへ飛ばします。**この処理は380行でやっています。

次に、見つかった場合、その要素番号を390行で変数ＯＲＤＥＲ１に代入しています。こうしておけば、630行からのＬＳＥＴ部分がそのまま使えますね。

そして、この代入された値をもとに、**400行、410行でデータが書き込まれているランダムファイルのセクタおよびレコード番号を算出し、420行でＧＥＴしています。**

これで、必要データがバッファ上に呼び出されましたから、このうち、**氏名だけを画面に表示して確認をします。**

プログラムは次のとおりです。

```
430 CLS
440 LOCATE 15,4 : PRINT NM$(ORDER2)
450 LOCATE 20,5 : PRINT "ｻﾝ ﾉ ｼｭｳｾｲ <y/n>?"
460 Y$=INKEY$ : IF Y$="" THEN GOTO 460
470 IF Y$="Y" OR Y$="y" THEN GOTO *CORRECT
480 IF Y$="N" OR Y$="n" THEN GOTO *LOOP
490 BEEP : BEEP :GOTO 460
```

画面に表示された人の修正で間違いなければ、470行で修正ルーチンへ飛ばしますし、間違っていれば、480行で、コード再入力へ飛ばして最初からやりなおしです。

490行はキー入力ミスの場合のフォローです。

データ修正

この部分の手順は、すでにＧＥＴ文でバッファ上に呼び出されている性別、氏名、郵便番号、住所、番地を次々に画面に表示し、必要があれば修正する、ということになります。

プログラムリストは少し長いですが、ＬＯＣＡＴＥ文、ＰＲＩＮＴ

●下のプログラムリストは行番号が２きざみで進む部分が多いですが、10きざみですと630行までの間に入りきりませんので急きょ、このようにしました。キーインがめんどうでしょうが、ヨロシク。

文、ＩＮＰＵＴ文だけですから、どうということはないでしょう。

次のようにキーインしますが、どのステートメントも形が同じなので、スクリーンエディタを使えば簡単ですよ。

たとえば、530行から534行をキーインし、これをベースにしてスクリーンエディタで536行から570行を作ればよいのです。変数名とＬＯＣＡＴＥ文のＹ座標を変更すればよいですね。あっ、そうそう、行番号を変えるのを忘れないでください。初めのうちはよく、こういったミスをおかすものです。

```
500 '
510 *CORRECT
520 CLS
530 LOCATE 15,4 : PRINT "ｼｭｳｾｲ ﾅｹﾚﾊﾞ RET"
532 LOCATE 15,6 : PRINT SEX$(ORDER1)
534 LOCATE 15,7 : INPUT "      ==> ";A1$
536 IF A1$="" THEN A1$=SEX$(ORDER1)
538 LOCATE 15,8 : PRINT NM$(ORDER2)
540 LOCATE 15,9 : INPUT "      ==> ";A2$
542 IF A2$="" THEN A2$=NM$(ORDER2)
544 LOCATE 15,10: PRINT PN$(ORDER2)
546 LOCATE 15,11: INPUT "      ==> ";A3$
548 IF A3$="" THEN A3$=PN$(ORDER2)
550 LOCATE 15,12: PRINT AD$(ORDER2)
552 LOCATE 15,13: INPUT "      ==> ";A4$
554 IF A4$="" THEN A4$=AD$(ORDER2)
556 LOCATE 15,14: PRINT BA$(ORDER2)
558 LOCATE 15,15: INPUT "      ==> ";A5$
560 IF A5$="" THEN A5$=BA$(ORDER2)
570 LOCATE 20,17 : PRINT "SURE <y/n> ?"
575 Y$=INKEY$ : IF Y$="" THEN GOTO 570
580 IF Y$="Y" OR Y$="y" THEN GOTO 630
585 IF Y$="N" OR Y$="n" THEN GOTO *CORRECT
590 BEEP : BEEP :GOTO 575
```

修正データの書き込み

上のプログラムをごらんください。

修正の必要がなければリターンキーが押されて、最初Ａ１＄などのストリング変数にヌルストリングが入りますが、この場合にはもう一度古いデータをこの変数に代入しなおしています。つまり、Ａ１＄～

A５＄にランダムファイルに書き込むべきデータが入っているわけですね。

こうしておくと、どうですか、46ページをごらんください。630行からのＬＳＥＴ文がそのまま使えますね。680行のＰＵＴ文も活用できます。そして、データをＰＵＴしたら、ふたたびコード入力に戻せばよいわけですから、690行もこのままでいいですね。

ということは、もうこれでプログラムが完成しました、と、ホッとしたいところですが、もう１つだけ万全を期しておきましょう。

修正する人が２人以上いる場合には、もう一度このルーチンに戻ってきます。そしてストリング変数Ａ１＄～Ａ５＄には、その前の文字列がそのまま残っています。

このように、ある変数を何回か使う場合には、一度クリアしておくようにしましょう。このプログラムではたぶんミスにはならないと思いますが、安全のためにステートメントを入れておきます。

書き込み部分のプログラムを次に示しますが、みなさんは682行だけキーインすればよいのですよ。

```
630 LSET SEX$(ORDER1)=A1$
640 LSET NM$(ORDER2) =A2$
650 LSET PN$(ORDER2) =A3$
660 LSET AD$(ORDER2) =A4$
670 LSET BA$(ORDER2) =A5$
680 PUT #1,SEC : PUT #2,1
682 A1$="" : A2$="" : A3$="" : A4$="" : A5$=""
690 GOTO *LOOP
```

キーインできましたら、最後にＲＥＭ文を入れておきましょう。

リストを次に示しますが、この部分はみなさんの好みに合わせてください。ただし、ファイル名だけは同じにして、おつきあいください。

それから、45ページのファイル名と比べてみると、スラッシュのうしろの数字が１か所異なるだけです。もし、いま使ったプログラムもここを「１」にしてセーブしたりすると、45ページのプログラムは消えてしまいますから注意してください〈注〉。

〈注〉このような事故を防ぐためにも、もう１つ別にバックアップファイルを作っておくとよいでしょう。そしてバックアップ作業はできるだけこまめに実行したほうがよいと思います。

```
10 ' file name ==> DM01/2.bas
20 ' made on 15th of JAN. 1984
30 ' random data correct
40 ' save "DM01/2.bas"
50 '
```

↑ 45ページのプログラム名とは、この数字が違うだけです

プログラムの実行

さて、プログラムが一応完成しましたので、リナンバリングする前に、正常に動くかどうかテストしてみたいと思いますが、まずプログラムをセーブしておいてください。

セーブできましたら、いよいよＲＵＮです。

（f・5）キーを押しますと、まずディスクをドライブ２にセットしたかどうかの確認メッセージが表示され、次にこのプログラムの見出しが現れます。

続いて、次のコード入力のメッセージです。

```
コード(END=999)?
```

ここでは「101」を入力してみましょう。キーイン後リターンキーを押します。

```
コード(END=999)? 101
```

すると、データサーチが行なわれ、カチ、カチという音とともに該当セクタをＧＥＴし、次のように氏名が表示されます。

```
AAAAAAA
     サン ノ シュウセイ <y/n>?
```

コード番号101は「ＡＡＡＡＡＡＡ」さんでしたね。間違いなさそうです。

もう一度ためしてみましょう。上の画面で「ｎ」をキーインし、コード入力画面に戻してください。そして今度はコード201を入力してみましょう。

すると、ピーッと音がして、次のように未登録のメッセージが表示されました。

```
コード(END=999)? 201

     トウロク シテ アリマセン!!
```

登録してないというのはおかしいですね。シーケンシャルファイルへの書き込みは間違いなくできているはずです。

どこが間違えているか調べてみましょう。

コード入力のメッセージが表示されたら、ストップキーを押してプログラムの処理を中断し、ダイレクトモードでＣＬＯＳＥをキーインしてください。

```
Break in 350   ←ストップキーを押す
Ok
close          ←ファイルをクローズする
Ok
■
```

次に、ダイレクトモードでシーケンシャルファイルをＯＰＥＮし、コードの書き込みが正確にできているか再確認してみます。

次の画面を見ながらキーインしてみてください。

```
open "2:INDX01.dat" for input as #1
Ok                                        ←ダイレクトモードでキーインする
for i=1 to 9:input #1,cd:? cd;:next i
 101  201  301  401  501  601  701  801  901
Ok
■                    データは正しく書き込まれている
```

データ書き込みは間違いなさそうですね。それでは配列の中味を調べてみましょう。まずＣＬＯＳＥをキーインし、ＲＵＮさせてコード入力でストップして配列をチェックします。

```
                 コード(END=999)?
Break in 350   ←ストップキーを押した
Ok
close
Ok
for i=1 to 9:? code(i);:next i   ←ダイレクトでキーイン
 101  301  501  701  901  0  0  0  0
Ok
■      ↑201が配列に読み込まれていない
```

おや？　おかしいですね。書き込んであるはずの201、401、601、801が表示されず、6番目から9番目まではすべて0となっています。この辺に問題解決のいとぐちがありそうです。

とにかく、配列内にコードが正しく読み込まれていないことがわかりました。では、この読み込みはどこで行なわれているのでしょうか。そうですね、3000行からのサブルーチンです。

50ページのリストを見てください。

3045行で、フラグが立っていなければ、つまり初回の読み込みでしたら、3050行へ飛ばしてカラ読みをします。

フラグが立っていれば3046行へ落ちてきますから、配列内にコードを読み込みます。しかし、そのあと3050行のＩＮＰＵＴ＃文も実行してしまいます。つまり、２回続けてＩＮＰＵＴ＃文を実行することになってしまいましたから、２つ目のデータは常にカラ読みされて、配列内には格納されなかったのです。

3046行のうしろにＧＯＴＯ文をつけ加えましょう。

```
3046    INPUT #3,CODE(COUNT) : GOTO 3060
```

今度は正常に動くと思います。

ＲＵＮさせて「501」の人の修正を行なってみましょう。

やり方はもうわかりますね。たとえば次のように、郵便番号、住所番地をなおしてみてください。

```
シュウセイ ナケレハ゛ RET

m
      ==> ?
EEEEEEE
      ==> ?
555
      ==> ? 123
eeeeeeeeeee
      ==> ? abcdefg
5-5-5
      ==> ? 1-2-3

      SURE <y/n> ?
```

修正データをキーイン

それでは「Ｙ」をキーインし、ランダムファイルに書き込みましょう。

ふたたび、コード入力メッセージが表示されますから、もう一度、501をキーインしてみてください。

なぜだかわかりますね？　正しく書き換えられたかどうか確認するためです。画面は次のようになります。

```
EEEEEEE
      サン ノ シュウセイ <y/n>?
```

「Y」を入力すると修正ルーチンに入りますが、修正なしということでリターンキーを押し続けると次のような画面になります。

```
シュウセイ ナケレバ゛ RET

m
      ==> ?
EEEEEEE
      ==> ?
123
      ==> ?
abcdefg
      ==> ?
1-2-3
      ==> ?

      SURE <y/n> ?
```

書き換えられたデータ

修正データはどれも、私たちの希望どおりに書き換えられていました。よかったですね。

プログラムがスピーディーに、それも正確に動いてくれるというのは、ほんとうにうれしいものです。現在の私はそんなふうに感じています。

こういった気持は誰でも、コンピュータを習いたての頃から変わらないのかもしれません。いつでしたか、あの、マシン語の神様といわれる川村清さんも同じようなことをいっていましたよ。

ホッとしたところでホットコーヒーを一杯といきたいところですが、まだリナンバリングが残っています。

アメリカンの味は、それまでおあづけ。

●リナンバリングしてセーブする前に45ページの注意書きをお読みください。

リナンバリングとセーブ

もう一息です。リナンバリングをして行番号を整えておきましょう。

```
renum 100,100
Ok
renum 1000,940
Ok
renum 2000,1070
Ok
renum 3000,2070
Ok
renum 4000,3110
Ok
■
```

キーインできましたら、一応リストを画面に表示させて、次にかかげるプログラム総リストとおおざっぱに照合してみてください。たとえば、サブルーチンの行番号などを比べてみればよいでしょう。

そして最後に、このプログラムのセーブです。

```
save "DM01/2.bas" ⏎
```

データ修正プログラムリスト

```
10 ' file name ==> DM01/2.bas
20 ' made on 15th of JAN. 1984
30 ' random data correct
40 ' save "DM01/2.bas"
50 '
100 CONSOLE 0,25,0,0 : WIDTH 80,25
110 DEFINT A-Z
120 DIM NM$(3),PN$(3),AD$(3),BA$(3)
130 DIM SEX$(256)
140 '
150 CLS
160 LOCATE 25,4 : PRINT "ﾄﾞﾗｲﾌﾞ2ﾍ ﾃﾞｨｽｸｦﾄﾞｳｿﾞ"
170 LOCATE 25,6 : PRINT "     Sure <y/n> ?"
180 Y$=INKEY$ : IF Y$="" THEN GOTO 180
```

```
190 IF Y$="Y" OR Y$="y" THEN GOTO 210
                        ELSE GOTO 160
200 '
210 CLS
220 GOSUB *OPNF01
230 GOSUB *OPNF02
240 GOSUB *READ.INDX : FLAG=1
250 DIM CODE(COUNT) : COUNT=0
260 GOSUB *READ.INDX
270 '
280 LOCATE 25,4 : PRINT "デ゛ータ ノ シュウセイ ヲ シマス"
290 FOR I=1 TO 3000 : NEXT I
300 '
310 *LOOP
320 CLS
330 LOCATE 15,4 : INPUT "コード゛(END=999)";CD
340 IF CD=999 THEN GOTO *FCLOSE
350 GOSUB *DAT.SEARCH
360 IF FOUND=-1 THEN GOTO *LOOP
370 ORDER1=FOUND
380 SEC=(ORDER1-1)¥3+2
390 ORDER2=(ORDER1-1) MOD 3+1
400 GET #1,SEC : GET #2,1
410 CLS
420 LOCATE 15,4 : PRINT NM$(ORDER2)
430 LOCATE 20,5 : PRINT "サン ノ シュウセイ <y/n>?"
440 Y$=INKEY$ : IF Y$="" THEN GOTO 440
450 IF Y$="Y" OR Y$="y" THEN GOTO *CORRECT
460 IF Y$="N" OR Y$="n" THEN GOTO *LOOP
470 BEEP : BEEP :GOTO 440
480 '
490 *CORRECT
500 CLS
510 LOCATE 15,4 : PRINT "シュウセイ ナケレハ゛ RET"
520 LOCATE 15,6 : PRINT SEX$(ORDER1)
530 LOCATE 15,7 : INPUT "      ==> ";A1$
540 IF A1$="" THEN A1$=SEX$(ORDER1)
550 LOCATE 15,8 : PRINT NM$(ORDER2)
560 LOCATE 15,9 : INPUT "      ==> ";A2$
570 IF A2$="" THEN A2$=NM$(ORDER2)
580 LOCATE 15,10: PRINT PN$(ORDER2)
590 LOCATE 15,11: INPUT "      ==> ";A3$
```

```
600 IF A3$="" THEN A3$=PN$(ORDER2)
610 LOCATE 15,12: PRINT AD$(ORDER2)
620 LOCATE 15,13: INPUT "      ==> ";A4$
630 IF A4$="" THEN A4$=AD$(ORDER2)
640 LOCATE 15,14: PRINT BA$(ORDER2)
650 LOCATE 15,15: INPUT "      ==> ";A5$
660 IF A5$="" THEN A5$=BA$(ORDER2)
670 LOCATE 20,17 : PRINT "SURE <y/n> ?"
680 Y$=INKEY$ : IF Y$="" THEN GOTO 670
690 IF Y$="Y" OR Y$="y" THEN GOTO 720
700 IF Y$="N" OR Y$="n" THEN GOTO *CORRECT
710 BEEP : BEEP :GOTO 680
720 LSET SEX$(ORDER1)=A1$
730 LSET NM$(ORDER2) =A2$
740 LSET PN$(ORDER2) =A3$
750 LSET AD$(ORDER2) =A4$
760 LSET BA$(ORDER2) =A5$
770 PUT #1,SEC : PUT #2,1
780 A1$="" : A2$="" : A3$="" : A4$="" : A5$=""
790 GOTO *LOOP
800 '
810 *FCLOSE
820 CLOSE #1,#2
830 '
840 CLS
850 LOCATE 25,4 : PRINT "ﾎｶ ﾉ ｼｺﾞﾄ ｦ ｼﾏｽｶ ?"
860 LOCATE 30,5 : PRINT "< y/n >"
870 Y$=INKEY$ : IF Y$="" THEN GOTO 870
880 IF Y$="Y" OR Y$="y" THEN RUN "DMMENU.bas"
890 IF Y$="N" OR Y$="n" THEN GOTO *FINISH
900 BEEP : BEEP :GOTO 850
910 '
920 *FINISH
930 END
1000 '
1010 *OPNF01
1020 OPEN "2:MAIL01.dat" AS #1
1030 FOR I=1 TO 3
1040   FIELD #1,85*(I-1) AS DUMMY1$,20 A
 S NM$(I),6 AS PN$(I),39 AS AD$(I),20 AS
 BA$(I)
1050 NEXT I
```

```
1060 RETURN
2000 '
2010 *OPNF02
2020 OPEN "2:MAIL01.dat" AS #2
2030 FOR I=1 TO 256
2040   FIELD #2,(I-1) AS DUMMY2$,1 AS SEX$(I)
2050 NEXT I
2060 RETURN
3000 '
3010 *READ.INDX
3020 OPEN "2:INDX01.dat" FOR INPUT AS #3
3030 WHILE NOT(EOF(3))
3040   COUNT=COUNT+1
3050   IF FLAG<>1 THEN GOTO 3070
3060   INPUT #3,CODE(COUNT) : GOTO 3080
3070   INPUT #3,DUMMY
3080 WEND
3090 CLOSE #3
3100 RETURN
4000 '
4010 *DAT.SEARCH
4020 FOUND=SEARCH(CODE,CD)
4030 IF FOUND<>-1 THEN RETURN
4040 LOCATE 20,6 : BEEP : BEEP
4050 PRINT "ﾄｳﾛｸ ｼﾃ ｱﾘﾏｾﾝ!!"
4060 FOR I=1 TO 3000 : NEXT I
4070 RETURN
```

バックアップディスクを作ってある方は、そちらにもこのプログラムをセーブしておいてください。どんな事故があるかわかりませんので……。念のため……。

1·7 宛て名の印刷

この節では、私たちがランダムファイルに登録した人たちの宛て名をディスプレイに表示、あるいはプリンタへ出力してみたいと思います。プリンタをお持ちでない方もいらっしゃると思いますので、どちらの方にも使えるように兼用のプログラムにするつもりです。

印刷までの手順

およそ次のような順序で処理を行ないます。

❶ 初期設定をしてからランダムファイルをＯＰＥＮ
❷ 全員への宛て名か女性だけかなどのメニューを表示して選択
❸ ランダムファイルの第１セクタのデータより必要人数を調べる
❹ 第２セクタ以降から必要データをＧＥＴして出力

❷で述べたメニューの中には当然、コード入力によって宛て名を出力する人を特定するケースも含まれますが、この節では、全員、男性だけ、女性だけの３通りの選択を考えます。

このプログラムにおいても、**ＯＰＥＮするランダムファイルが前回のプログラムと共通なので、データ修正のプログラムの中で流用できる部分があります。**

既存のプログラムの一部を流用する方法は前節でも実践しましたが、利用できるものは利用するという考え方に立ったほうが、キーインの手間もはぶけますし、少なくともその部分でのバグはまぬがれますから、かなりのメリットがあると思います。

メニュー選択までのプログラム

おそれいりますが、60ページから63ページまでのプログラムリストをごらんください。

先頭のリマーク文は、スクリーンエディタで修正することにすれば、10行から、ランダムファイルＯＰＥＮへのＧＯＳＵＢ文である230行までは、そのまま使えます。

240行から790行まではデータ修正ルーチンですので削除します。そして、この部分に、メニュー選択および印刷ルーチンをはめ込みます。

800行から2060行まではそのまま流用、3000行および4000行からの

サブルーチンは不要になります。

60～63ページのリストでは4ページにまたがりますので、必要なステートメント、削除する行を一度で把握するというわけにはいかないと思いますが、プリンタをお持ちでしたらＬＬＩＳＴコマンドでプリンタに出力したリストをごらんになると、プログラムの流れがひと目でわかって理解しやすいと思います。

では、60ページのデータ修正プログラムをロードしてから、次のようにダイレクトモードで不要行をデリートします。

```
delete 240-790
Ok
delete 3000-4070
Ok
■
```

さあ、これで半分以上プログラムが短くなりました。あとは、必要に応じて、ＬＩＳＴコマンドでリストを画面に表示させながら読み進んでください。

まず先頭のリマーク文をなおしておきましょう。

```
10 ' file name ==> DM01/3.bas
20 ' 1984.1.20
30 ' random data print out
40 ' save "DM01/3.bas"
50 '
```

上のリストで、緑色の個所が修正点です。

行番号40は、この新しいプログラムをセーブするときに使えますから、早めになおしておくと、前のプログラムと混同せず、大変便利です。

次は、プリンタを持っているかどうかのチェックをするステートメントを作っておきます。

```
240 '
250 LOCATE 25,4 : PRINT "ﾌﾟﾘﾝﾀ ﾊ <y/n> ?"
260 Y$=INKEY$ : IF Y$="" THEN GOTO 260
270 IF Y$="Y" OR Y$="y" THEN P.FLAG=1
```

プリンタを持っている場合には、270行のTHEN文以降で、プリンタ所持判定用のフラグを立てて（1にして）います。このフラグは、あとで、宛て名をディスプレイ出力かプリンタ出力かの判定に使用します。

キーインできましたら次は、メニュー表示のステートメントです。メニュー画面のレイアウトは次のように考えてみました。

```
メニュー ヲ ト゛ウソ゛

セ゛ンイン・・・・・・1
ダ゛ンセイ・・・・・・2
シ゛ョセイ・・・・・・3
トクテイ ノ ヒト・・4

Input No.? ■
```

メニューは4つですね。

登録してある人全員へ、男性のみ、女性のみ、それに特定の人への宛て名印刷です。

プリンタを持っているかどうかの確認が終わったら、メニューステートメントを続けます。

次のようにキーインしてください。LOCATE位置は、あとでみなさんが調整してください。

```
280 '
290 CLS
300 LOCATE 25,4  : PRINT "メニュー ヲ ト゛ウソ゛"
310 LOCATE 25,6  : PRINT "セ゛ンイン・・・・・・1"
320 LOCATE 25,7  : PRINT "ダ゛ンセイ・・・・・・2"
330 LOCATE 25,8  : PRINT "シ゛ョセイ・・・・・・3"
340 LOCATE 25,9  : PRINT "トクテイ ノ ヒト・・4"
350 LOCATE 25,11 : INPUT "Input No.";NO
```

次は、選択された各メニュー番号に対して、それぞれの処理へ分岐させる部分です。プログラムを次に示します。

```
360 IF NO=1 THEN GOSUB *ZENIN : GOTO *PRT
370 IF NO=2 THEN A$="m" : B$="M" : GOSUB *MF
    : GOTO *PRT
380 IF NO=3 THEN A$="f" : B$="F" : GOSUB *MF
    : GOTO *PRT
390 IF NO=4 THEN RUN "DM01/4.bas"
400 BEEP : BEEP ;GOTO 350
```

まず、メニュー番号１が選択された場合には、サブルーチンに飛ばし、2000行からのサブルーチンですでにＯＰＥＮされているランダムファイルの第１セクタをＧＥＴし、書き込まれているデータ数をチェックします。つまり、全部で何人いるかを調べるわけです。

そしてチェックが終わると、踏み台になったＧＯＳＵＢ文のすぐうしろに戻ってきますから、続くＧＯＴＯ文を実行し、宛て名出力部分へジャンプさせます。

メニュー番号２と３を選んだときも、考え方はまったく同じです。ただ、女性か男性かの判定をする必要がありますので、メニュー番号１の場合とは別のサブルーチンを作ります。

370では、変数Ａ＄、Ｂ＄に小文字と大文字のエムを代入していますが、これは、データ書き込み時に、あやまって大文字を入力していることも予想されますので、そのための処置です。

メニュー番号１から３までは、別にコード入力の必要がありませんでしたが、メニュー番号４の特定の人の宛て名はコード入力によりますのでデータサーチの部分を追加しなければなりません。しかし、データサーチはデータ修正プログラムの該当個所がほぼそのまま使えますので、この場合は専用のプログラム「ＤＭ01／４.bas」を作り、これを実行させることにしました。

メニュー番号４を選択した場合の処理も、いま私たちが作っているプログラムに組み込むこともできますが、リストがそれだけ長くなり、あとでリストをチェックするときなど大変不便です。

また、プログラムリストをあまり長くすると、私たちの使えるフリーエリアが少なくなり、データをたくさん扱うようになると、メモリ不足になったりします。

最近ディスクの容量はますますふえています。データファイルでは有効に使ったほうがよいと思いますが、プログラムファイルではディスクの効率はあまり気にせず、できたら処理別にこざっぱりと分けて、チェインとかＲオプション付きのロードで操作したほうが、ジョブシステムを管理する上で大変有利だと思います。

長いプログラムを作っていたのは、カセートテープ使用時の話で、いまはディスクという強力な道具があります。余裕ができたのですから、プログラムにもゆとりを持たせて、ゆったり使いたいものです。要は、私たちが管理する際、できるだけ簡潔でわかりやすくしておくことだと思います。

サブルーチンを作る

話が脱線しましたが、2つのサブルーチンを完成させておきましょう。

まず、メニュー番号1が選択された場合のサブーチンから。

```
3000 '
3010 *ZENIN
3020 GET #2,1
3030 FOR I=1 TO 256
3040   IF SEX$(I)="m" OR SEX$(I)="M" THEN
        COUNT=COUNT+1 ELSE IF SEX$(I)="f"
        OR SEX$(I)="F" THEN COUNT=COUNT+1
3050 NEXT I
3060 RETURN
```

3020行で、ランダムファイルの第1セクタをＧＥＴし、データをバッファ上に取り出します。そして、3030行から3050行で256回ループさせて、このデータが男性と女性のチェッカーかどうかを調べます。

男性でも女性でもなければ、カウントされず、ＮＥＸＴ文へ落ちていきます。

それではここで、このサブルーチンがうまく動作するかどうかテストをしてみたいと思います。

プログラム作成中は、できるだけこまめにテストを行なったほうがよいと思います。というのは、完成してからテストとなりますと、プログラムもかなり長くなりますし、エラーとかバグがでた場合、いろいろなルーチンとの関連の上でのエラーですから、探すのにひと苦労します。

それでは簡単なテストをしてみますが、360行に、次ページに示すリストのように、ＰＲＩＮＴ文とＥＮＤ文を挿入してください。

カウンタ用変数の中味が何かを調べ、正確に登録データ数をカウントできたかどうかチェックするわけです。

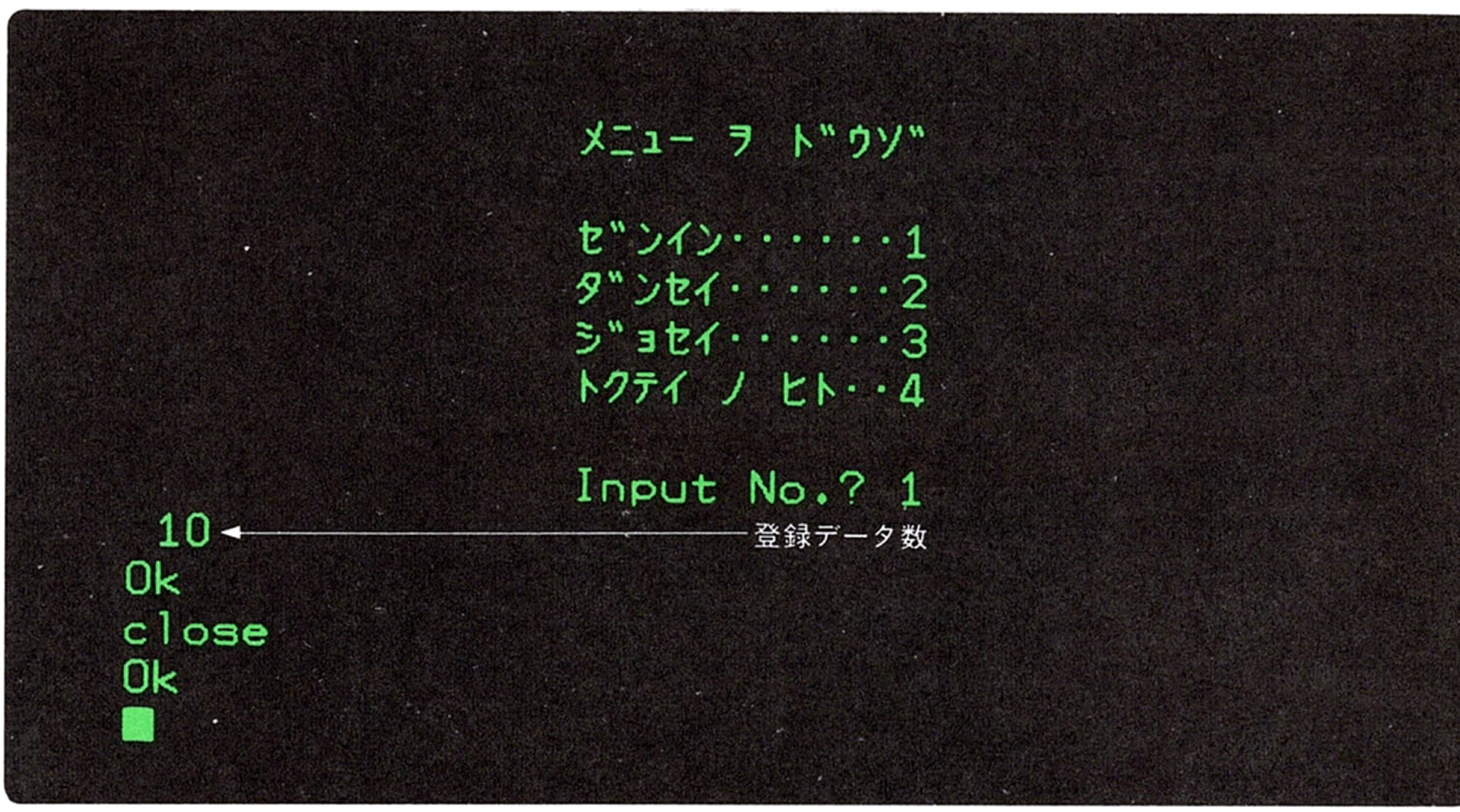
```
360 IF NO=1 THEN GOSUB *ZENIN : PRINT COUNT
    : END : GOTO *PRT
```

緑色の個所が挿入したステートメントです。

(f・5)キーでRUNしますと、ドライブ2へデータファイル用のディスクをセットしたかどうかたずねてきますから答えてください。次に、プリンタを持っているかどうか聞いてきます。YかNを入力してください。

すると今度はメニュー画面です。ここでは「1」をキーインしてください。ほかの番号だとエラーになりますよ。

画面は次のようになりました。



ここで一応CLOSEをキーインし、リターンキーを押しておいてください。一応と申しましたのは、(360行にあります) END文は自動的にファイルをクローズしてくれますのでわざわざキーインすることはないのですが、習慣づけておいたほうがよいと思うからです。私も以前、ついうっかりCLOSEをおこたったためにエライ目にあった経験があります。

上の画面を見てください。「10」という数字が表示されています。私たちが書き込んだのは10件でしたね。間違いなさそうです。

それでは、360行をもとどおりになおしておきましょう。

```
360 IF NO=1 THEN GOSUB *ZENIN : GOTO *PRT
```

それにしてもちょっと処理速度が遅いですね。そして、なんとかならないものかと考えているうちにハッと思ったのですが、3000行からのサブルーチンの目的は登録してあるレコード数を知るだけですから、シーケンシャルファイルを使えば、もっとすみやかに処理できるのでした。

プログラムを完成させてから、この原稿を書いているのではなく、プログラミングと同時進行でやっておりますので、こんなことになってしまいました。

まあ、先ほど入力したサブルーチンも間違いではなく、多少時間がかかるというだけですから、勉強のためと思ってご容赦ください。

新しいサブルーチンを次に示します。どちらを使ってもかまいません。この本では、次のステートメントを採用します。

```
3000 '
3010 *ZENIN
3020 OPEN "2:INDX01.dat" FOR INPUT AS #3
3030 WHILE NOT(EOF(3))
3040   COUNT=COUNT+1
3050   INPUT #3,DUMMY
3060 WEND
3070 CLOSE #3
3080 RETURN
```

次は、370行、380行のＧＯＳＵＢ文の飛び先です。行番号を4000からとして、次のようにキーインしてください。

```
4000 '
4010 *MF
4020 GET #2,1
4030 FOR I=1 TO 256
4040   IF SEX$(I)=A$ OR SEX$(I)=B$
         THEN COUNT=COUNT+1 : CO(COUNT)=I
4050 NEXT I
4060 RETURN
```

やはり先ほどと同じように。ランダムファイルの第１セクタをＧＥＴし、Ａ＄、Ｂ＄の中にしまわれているチェッカーと比較して、もし該当するものであればカウンタを１つ上げて、そのときのループ変数Ｉの値（これが何番目のレコードかを示しています）を配列に格納しています。

このサブルーチンでは新しい配列を使っていますので、配列宣言をしておきましょう。

```
135 DIM CO(256)
```

このサブルーチンも正常に動くかどうか、69ページと同じ方法でテストしてみてください。

宛て名印刷ルーチン

ＬＩＳＴコマンドでプログラムリストを画面に表示させ、どの程度まで完成したかプログラムの流れをつかんでおいてください。

プログラミング中は、こまめにこのような操作をくりかえすとよいと思います。

さて、それでは印刷ルーチンの入口から。次のプログラムを入力してください。

```
410 '
420 *PRT
430 CLS
440 IF P.FLAG=1 THEN GOSUB *CHECK : GOTO 500
450 LOCATE 25,4 : PRINT "ｱﾃﾅ ｦ ﾋｮｳｼﾞｼﾏｽ"
460 FOR I=1 TO 5000 : NEXT I
```

440行のステートメントは、65ページのリストの250行以降でプリンタ所持の有無を確認していますが、持っていればフラグが立っていますから、サブルーチンでプリンタのセットをうながします。

```
5000 '
5010 *CHECK
5020 LOCATE 25,4 : PRINT "ﾌﾟﾘﾝﾀ ｦ ｾｯﾄｼﾃｸﾀﾞｻｲ"
5030 LOCATE 30,6 : PRINT "Sure < y/n >"
5040 Y$=INKEY$ : IF Y$="" THEN GOTO 5040
5050 IF Y$="Y" OR Y$="y" THEN GOTO 5080
5060 IF Y$="N" OR Y$="n" THEN GOTO 5040
5070 BEEP : BEEP : GOTO 5040
5080 CLS
5090 LOCATE 25,4 : PRINT "ｱﾃﾅ ｦ ｲﾝｻﾂ ｼﾏｽ"
5100 RETURN
```

5000行からのサブルーチンはむずかしいところはないので説明は省略します。

次に示すプログラムは、宛て名をディスプレイに表示、あるいはプリンタに印刷する部分です。

```
500 FOR I=1 TO COUNT
510   IF CO(I)=0 THEN ORDER1=I : GOTO 530
520   ORDER1=CO(I)
530   SEC=(ORDER1-1)¥3+2
540   ORDER2=(ORDER1-1) MOD 3+1
550   GET #1,SEC
560   IF P.FLAG=1 THEN GOTO 660
570   CLS
580   LOCATE 20,4 : PRINT NM$(ORDER2)
590   LOCATE 20,6 : PRINT PN$(ORDER2)
600   LOCATE 20,8 : PRINT AD$(ORDER2)
610   LOCATE 20,10: PRINT BA$(ORDER2)
620   LOCATE 25,12
630   PRINT "Hit RET key for next person!!"
640   Y$=INKEY$ : IF Y$="" THEN GOTO 640
650   GOTO 720
660   LPRINT PN$(ORDER2)
670   LPRINT AD$(ORDER2)
680   LPRINT BA$(ORDER2)
690   LPRINT TAB(5);NM$(ORDER2)
700   LPRINT TAB(15);"ｻﾏ"
710   FOR J=1 TO 5 : LPRINT : NEXT J
720 NEXT I
```

全データを印刷するとき（510行）
登録順位の代入（520行）
セクタ計算（530行）
セクタ内のレコード番号を計算（540行）
セクタをＧＥＴ（550行）
プリンタを持っていればジャンプ（560行）
ディスプレイに表示（580～610行）
プリンタで印刷（660～700行）
５行分だけ紙送りをする（710行）

コメントを入れておきましたので、ご理解いただけると思いますが、510行は、**すべてのデータを印刷する場合、配列ＣＯは使っていませんでしたので、その中味はすべて０**ですね。ですからこの場合には、ループ変数Ｉの値を該当セクタおよびセクタ内でのレコード番号を算出する変数ＯＲＤＥＲ１に代入しています。

一方、**男女別に印刷するときは、配列ＣＯの中に、書き込んであるデータのトップを１番とする登録順位が入っていますから、これを同じくＯＲＤＥＲ１に代入します。**

データの書き込まれているセクタおよびセクタ内でのレコード番号の計算は、前に勉強しましたのでわかると思います。530、540行がそうですね。

あとは、そのセクタをＧＥＴし、ディスプレイあるいはプリンタに

出力し、この処理を変数ＣＯＵＮＴにしまわれている回数だけくりかえせばよいわけです。

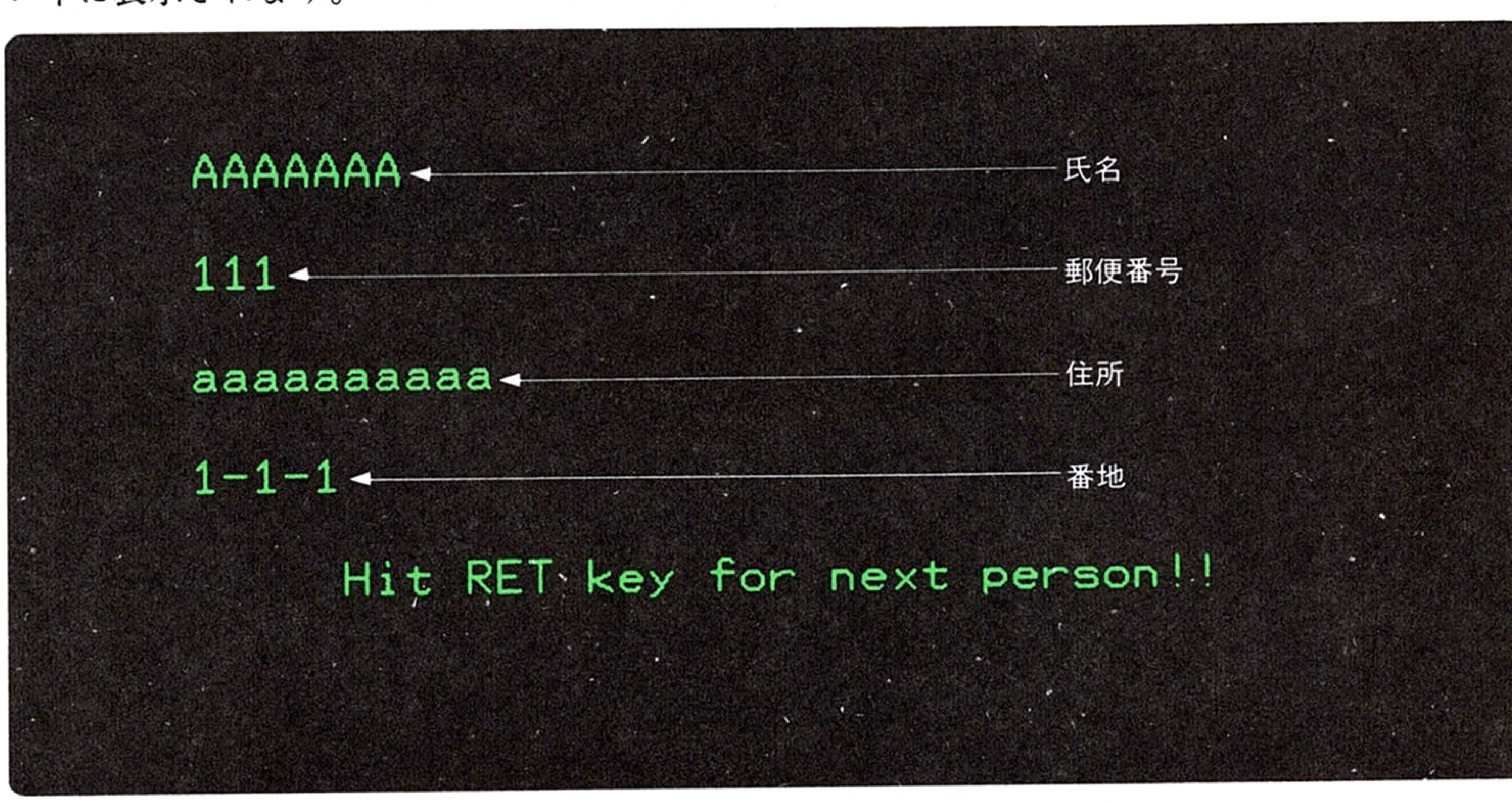
プログラムの実行

これでプログラムが完成しましたので実行してみます。

（f・5）キーを押してＲＵＮさせると、最初ディスクをドライブ２にセットしたかどうかの確認メッセージ、続いてプリンタの有無をたずねてきます。

「Ｎ」「Ｙ」いずれをキーインしても、66ページのメニュー画面が表示されますが、ここでは「Ｎ」をキーインし、続いてメニュー番号２を選択してみましょう。

しばらくすると、次のように、第１番目の男性のデータがディスプレイに表示されます。



以後、リターンキーを押すたびに登録データの中から男性のみを表示します。

次に、もう一度ＲＵＮさせて、データをプリンタに出力してみましょう。

今度はメニュー番号３を選択して女性のデータを印刷してみます。プリンタ出力の場合には、メニュー選択後、プリンタセットの確認メッセージが表示されます。

電源ＯＮにして、ＳＥＬスイッチを入れたかどうか確かめてください。「Ｙ」を入力すると印刷が開始され、女性５人分のデータが印刷されます。次ページに最初の２名分の出力例を示します。

```
333
cccccccccc
3-3-3
          CCCCCCC
                    サマ




444
dddddddddd
4-4-4
          DDDDDDD
                    サマ
```

上の出力例を見ますと、「サマ」を氏名のうしろに続けて表示しないで、１行改行してプリントしていますが、これは、プログラムリストの1040行で氏名用変数ＮＭ＄に20バイトのフィールドを割り当てていますので、たとえば氏名が11文字でおさまるような人の場合、「サマ」を続けて出力しようとしても、氏名のうしろに９字分のスペースが入りますので、ちょっと間が抜けた感じになります。

ですから、氏名の平均的な長さを考慮して、１行改行してから「サマ」を出力すれば、まとまった形にすることができます。

プログラムリスト

それではリナンバリングしてセーブしておきましょう。

```
renum 100,100
renum 1000,850
renum 2000,1070
renum 3000,2070
renum 4000,3090
renum 5000,4070
```

```
10 ' file name ==> DM01/3.bas
20 ' 1984.1.20
30 ' random data print out
40 ' save "DM01/3.bas"
50 '
100 CONSOLE 0,25,0,0 : WIDTH 80,25
110 DEFINT A-Z
120 DIM NM$(3),PN$(3),AD$(3),BA$(3)
130 DIM SEX$(256)
140 DIM CO(256)
150 '
160 CLS
170 LOCATE 25,4 : PRINT "ﾄﾞﾗｲﾌﾞ2ﾍ ﾃﾞｨｽｸｦﾄﾞｳｿﾞ"
180 LOCATE 25,6 : PRINT "     Sure <y/n> ?"
190 Y$=INKEY$ : IF Y$="" THEN GOTO 190
200 IF Y$="Y" OR Y$="y" THEN GOTO 220
                        ELSE GOTO 170
210 '
220 CLS
230 GOSUB *OPNF01
240 GOSUB *OPNF02
250 '
260 LOCATE 25,4 : PRINT "ﾌﾟﾘﾝﾀ ﾊ <y/n> ?"
270 Y$=INKEY$ : IF Y$="" THEN GOTO 270
280 IF Y$="Y" OR Y$="y" THEN P.FLAG=1
290 '
300 CLS
310 LOCATE 25,4  : PRINT "ﾒﾆｭｰ ｦ ﾄﾞｳｿﾞ"
320 LOCATE 25,6  : PRINT "ｾﾞﾝｲﾝ······1"
330 LOCATE 25,7  : PRINT "ﾀﾞﾝｾｲ······2"
340 LOCATE 25,8  : PRINT "ｼﾞｮｾｲ······3"
350 LOCATE 25,9  : PRINT "ﾄｸﾃｲ ﾉ ﾋﾄ··4"
360 LOCATE 25,11 : INPUT "Input No.";NO
370 IF NO=1 THEN GOSUB *ZENIN : GOTO *PRT
380 IF NO=2 THEN A$="m" : B$="M" : GOSUB *MF
    : GOTO *PRT
390 IF NO=3 THEN A$="f" : B$="F" : GOSUB *MF
    : GOTO *PRT
400 IF NO=4 THEN RUN "DM01/4.bas"
410 BEEP : BEEP ;GOTO 360
420 '
430 *PRT
440 CLS
```

```
450 IF P.FLAG=1 THEN GOSUB *CHECK : GOTO 480
460 LOCATE 25,4 : PRINT "ｱﾃﾅ ｦ ﾋｮｳｼﾞｼﾏｽ"
470 FOR I=1 TO 5000 : NEXT I
480 FOR I=1 TO COUNT
490   IF CO(I)=0 THEN ORDER1=I : GOTO 510
500   ORDER1=CO(I)
510   SEC=(ORDER1-1)¥3+2
520   ORDER2=(ORDER1-1) MOD 3+1
530   GET #1,SEC
540   IF P.FLAG=1 THEN GOTO 640
550   CLS
560   LOCATE 20,4 : PRINT NM$(ORDER2)
570   LOCATE 20,6 : PRINT PN$(ORDER2)
580   LOCATE 20,8 : PRINT AD$(ORDER2)
590   LOCATE 20,10: PRINT BA$(ORDER2)
600   LOCATE 25,12
610   PRINT "Hit RET key for next person!!"
620   Y$=INKEY$ : IF Y$="" THEN GOTO 620
630   GOTO 700
640   LPRINT PN$(ORDER2)
650   LPRINT AD$(ORDER2)
660   LPRINT BA$(ORDER2)
670   LPRINT TAB(5);NM$(ORDER2)
680   LPRINT TAB(15);"ｻﾏ"
690   FOR J=1 TO 5 : LPRINT : NEXT J
700 NEXT I
710 '
720 *FCLOSE
730 CLOSE #1,#2
740 '
750 CLS
760 LOCATE 25,4 : PRINT "ﾎｶ ﾉ ｼｺﾞﾄ ｦ ｼﾏｽｶ ?"
770 LOCATE 30,5 : PRINT "< y/n >"
780 Y$=INKEY$ : IF Y$="" THEN GOTO 780
790 IF Y$="Y" OR Y$="y" THEN RUN "DMMENU.bas"
800 IF Y$="N" OR Y$="n" THEN GOTO *FINISH
810 BEEP : BEEP :GOTO 760
820 '
830 *FINISH
840 END
1000 '
1010 *OPNF01
1020 OPEN "2:MAIL01.dat" AS #1
```

```
1030 FOR I=1 TO 3
1040   FIELD #1,85*(I-1) AS DUMMY1$,20 A
 S NM$(I),6 AS PN$(I),39 AS AD$(I),20 AS
 BA$(I)
1050 NEXT I
1060 RETURN
2000 '
2010 *OPNF02
2020 OPEN "2:MAIL01.dat" AS #2
2030 FOR I=1 TO 256
2040   FIELD #2,(I-1) AS DUMMY2$,1 AS SEX$(I)
2050 NEXT I
2060 RETURN
3000 '
3010 *ZENIN
3020 OPEN "2:INDX01.dat" FOR INPUT AS #3
3030 WHILE NOT(EOF(3))
3040   COUNT=COUNT+1
3050   INPUT #3,DUMMY
3060 WEND
3070 CLOSE #3
3080 RETURN
4000 '
4010 *MF
4020 GET #2,1
4030 FOR I=1 TO 256
4040   IF SEX$(I)=A$ OR SEX$(I)=B$
         THEN COUNT=COUNT+1 : CO(COUNT)=I
4050 NEXT I
4060 RETURN
5000 '
5010 *CHECK
5020 LOCATE 25,4 : PRINT "ﾌﾟﾘﾝﾀ ｦ ｾｯﾄｼﾃｸﾀﾞｻｲ"
5030 LOCATE 25,6 : PRINT "  Sure < y/n > ?"
5040 Y$=INKEY$ : IF Y$="" THEN GOTO 5040
5050 IF Y$="Y" OR Y$="y" THEN GOTO 5080
5060 IF Y$="N" OR Y$="n" THEN GOTO 5040
5070 BEEP : BEEP : GOTO 5040
5080 CLS
5090 LOCATE 25,4 : PRINT "ｱﾃﾅ ｦ ｲﾝｻﾂ ｼﾏｽ"
5100 RETURN
```

1・8　コード入力による宛て名印刷

ここで作るプログラムは、前節のメニューで「4」を選択した場合実行して処理するものですが、前節のプログラムおよびその前のデータ修正プログラムの両方にそのまま使える部分がありますので、

DM01/2.bas―――――❶
DM01/3.bas―――――❷

の2つのプログラムの不要部分を削除して、お互いにドッキングさせて1つプログラムに仕上げたいと思います。

あるプログラムに他のプログラムを結合させるにはMERGEを使いますが、この節では、キー入力の手間をはぶき、楽をしてプログラムを作り上げる方法を学んでいきましょう。

ベースにするファイルは❶を使い、このプログラムでは初期設定、ファイルOPEN、データサーチの部分を生かします。

また、ドッキングさせるほうのプログラムには❷を使い、こちらは宛て名の印刷部分のみを生かして使うことにします。

ここでの処理のおおまかな流れは、

1――コード入力
2――データの確認
3――データの登録
4――データの印刷

となりますから、これを頭において読み進めてください。

ベースのプログラムを作る

まず、❶のプログラムをロードしてください。

```
load "DM01/2.bas" ⏎
```

ロードできましたか？

それでは不要なステートメントを検討してみましょう。

140行から190行までは、ディスクをドライブ2にセットしたかどうかの確認ですから、これは前節のプログラムの実行で行なわれていますから必要ありません。

次に、480行から790行までも、修正ルーチンですので不要ですね。

あとは、多少手なおしする程度で、だいたい使えそうです。

では不要部分をデリートしておきましょう。

```
delete 140-190
Ok
delete 480-790
Ok
■
```

キーインできましたら、リストコマンドで一応確かめておいてください。

次に、先頭のREM文を修正しておきます。緑色の部分が修正個所です。

```
10 ' file name ==> DM01/4.bas
20 ' 1984.1.20
30 ' random data print out through code
40 ' save "DM01/4.bas"
50 '
```

ここではデリート後のプログラムリストを示しませんので、60～63ページのリストを見ながらお願いします。

260行ではサブルーチンへジャンプして配列にコードを読み込んできますが、戻ってきたら、カウンタ用変数はゼロクリアしておきましょう。あとで、登録数をカウントするのにもう一度使います。

```
260 GOSUB *READ.INDX : COUNT=0
```

リストコマンドで260行を表示し、**スクリーンエディタで修正したら、必ずリターンキーを押してください。**

次は、280行のメッセージをなおしておきます。

```
280 LOCATE 25,4 : PRINT "ﾃﾞｰﾀ ﾉ ｲﾝｻﾂ ｦ ｼﾏｽ"
```

340行は、コードに「999」がキーインされたら、終りの処理へ飛ばしていますが、このプログラムではデータをディスプレイまたはプリンタに出力するルーチンへジャンプさせなければなりませんね。

次のように修正しておきましょう。

```
340 IF CD=999 THEN GOTO *PRT
```

430行のメッセージも変えておきます。

```
430 LOCATE 20,5 : PRINT "サン ノ インサツ <y/n>?"
```

このメッセージに対して「Y」を入力すれば、450行で登録ルーチンへ飛ばすことにしますので、ジャンプ先を変更しておきます。

```
450 IF Y$="Y" OR Y$="y" THEN GOTO *TOROKU
```

そして次に、登録する場合のルーチンを完成させておきます。

```
500 '
510 *TOROKU
520 COUNT=COUNT+1 : CO(COUNT)=FOUND
530 GOTO *LOOP
```

この場合、**登録するごとにカウンタを１つ上げ、レコード番号を配列ＣＯの中にしまっておきます。**

あとはふたたびコード入力に飛ばせばよいですね。

これでベースとなるプログラムの修正は終りました。

新しいファイル名でセーブしておきましょう。

```
save "DM01/4.bas"
```

この数字を間違わないようにしてください

それでは私たちの次のテーマ、マージするプログラムを作ろうと思いますが、説明をするのに便利ですので、ベースとなるプログラムの行番号200から800までを右ページにかかげておきます。

マージするプログラムを作る

初めに78ページで述べた❷のプログラムをロードしてください。

```
load "DM01/3.bas"
```

そうしましたら、前と同じように不要行を削除しますが、75～77ページのリストをごらんください。

10～240行は初期設定部とファイルＯＰＥＮですので、ベースプログラムと重複しますのでデリートします。

また、300～410行のメニュー画面も不要ですね。

それから490行も、これは全データをプリントするときのステート

```
200 '
210 CLS
220 GOSUB *OPNF01
230 GOSUB *OPNF02
240 GOSUB *READ.INDX : FLAG=1
250 DIM CODE(COUNT) : COUNT=0
260 GOSUB *READ.INDX : COUNT=0
270 '
280 LOCATE 25,4 : PRINT "デ゛ータ ノ インサツ ヲ シマス"
290 FOR I=1 TO 3000 : NEXT I
300 '
310 *LOOP
320 CLS
330 LOCATE 15,4 : INPUT "コート゛(END=999)";CD
340 IF CD=999 THEN GOTO *PRT
350 GOSUB *DAT.SEARCH
360 IF FOUND=-1 THEN GOTO *LOOP
370 ORDER1=FOUND
380 SEC=(ORDER1-1)¥3+2
390 ORDER2=(ORDER1-1) MOD 3+1
400 GET #1,SEC : GET #2,1
410 CLS
420 LOCATE 15,4 : PRINT NM$(ORDER2)
430 LOCATE 20,5 : PRINT "サン ノ インサツ <y/n>?"
440 Y$=INKEY$ : IF Y$="" THEN GOTO 440
450 IF Y$="Y" OR Y$="y" THEN GOTO *TOROKU
460 IF Y$="N" OR Y$="n" THEN GOTO *LOOP
470 BEEP : BEEP :GOTO 440
500 '
510 *TOROKU
520 COUNT=COUNT+1 : CO(COUNT)=FOUND
530 GOTO *LOOP
800 '
```

⬆プログラムリストA

メントですから不要です。

さらに、710～4060もベースのプログラムと重複するのでいりません。

削除するステートメントは、だいたい以上の通りだと思います。

さっそくダイレクトモードでキーインし、不要な行をデリートしておきましょう。

```
delete 10-240
Ok
delete 290-410
Ok
490
delete 710-4060
Ok
■
```

マージ用のプログラムにするには、もう少し細工をしなければなりませんので、デリート後のリストのうち、250行から5100行までを次に示します。

⬇プログラムリストB

```
250 '
260 LOCATE 25,4 : PRINT "ﾌﾟﾘﾝﾀ ﾊ <y/n> ?"
270 Y$=INKEY$ : IF Y$="" THEN GOTO 270
280 IF Y$="Y" OR Y$="y" THEN P.FLAG=1
420 '
430 *PRT
440 CLS
450 IF P.FLAG=1 THEN GOSUB *CHECK : GOTO 480
460 LOCATE 25,4 : PRINT "ｱﾃﾅ ｦ ﾋｮｳｼﾞｼﾏｽ"
470 FOR I=1 TO 5000 : NEXT I
480 FOR I=1 TO COUNT
500   ORDER1=CO(I)
510   SEC=(ORDER1-1)¥3+2
520   ORDER2=(ORDER1-1) MOD 3+1
530   GET #1,SEC
540   IF P.FLAG=1 THEN GOTO 640
550   CLS
560   LOCATE 20,4 : PRINT NM$(ORDER2)
570   LOCATE 20,6 : PRINT PN$(ORDER2)
580   LOCATE 20,8 : PRINT AD$(ORDER2)
590   LOCATE 20,10: PRINT BA$(ORDER2)
600   LOCATE 25,12
610   PRINT "Hit RET key for next person!!"
620   Y$=INKEY$ : IF Y$="" THEN GOTO 620
630   GOTO 700
640   LPRINT PN$(ORDER2)
```

この部分をリストAの300行と310行の間に入れたい（260〜280行）

この部分はリストAの530行よりうしろに入れたい（420〜640行）

```
650    LPRINT AD$(ORDER2)
660    LPRINT BA$(ORDER2)
670    LPRINT TAB(5);NM$(ORDER2)
680    LPRINT TAB(15);"ｻﾏ"
690    FOR J=1 TO 5 : LPRINT : NEXT J
700 NEXT I
5000 '
5010 *CHECK
5020 LOCATE 25,4 : PRINT "ﾌﾟﾘﾝﾀ ｦ ｾｯﾄｼﾃｸﾀﾞｻｲ"
5030 LOCATE 25,6 : PRINT "  Sure < y/n > ?"
5040 Y$=INKEY$ : IF Y$="" THEN GOTO 5040
5050 IF Y$="Y" OR Y$="y" THEN GOTO 5080
5060 IF Y$="N" OR Y$="n" THEN GOTO 5040
5070 BEEP : BEEP : GOTO 5040
5080 CLS
5090 LOCATE 25,4 : PRINT "ｱﾃﾅ ｦ ｲﾝｻﾂ ｼﾏｽ"
5100 RETURN
```

上のリストをごらんください。

このプログラムをマージして、先ほどセーブした「DM01／4.bas」と結合させるわけですが、ＭＥＲＧＥの場合には、２つのプログラムの行番号が同じときは、あとからつなげるほうの行番号が優先され、ベースとなるプログラムのそれは消えてしまいます。

私たちはいま、上のリストのプリンタ有無の確認部分である250行から280行はリストＡの300行と310行の間に、420行から700行までのデータ出力の部分はリストＡの530行と800行の間に入れたいと考えています。

としますと、リストＡとリストＢでは行番号の重複する部分がありますから、これを調整しておかなければなりません。

ＲＥＮＵＭコマンドで、リストＢの行番号を次のようにつけかえておきましょう。

```
renum 301,250,1
Ok
renum 600,305,5
Ok
renum 5000,740
Ok
█
```

リストBの行番号250を301とし1きざみに、同じくリストBの行番号420を600行として5きざみにし、サブルーチンは5000行からとします。

ダイレクトモードでRENUMコマンドを実行し、リナンバリングできましたら、LLSTコマンドで一応確認しておいてください。

間違いなかったら、このプログラムを次のようにアスキーセーブしておきます。

```
save "DMmrg.asc",a
```

↑ アスキーセーブ

そしてマージ

いよいよ2つのプログラムのドッキングです。

まず、ベースとなるプログラムをロードし、次にMERGEコマンドで先ほどアスキーセーブしたプログラムをつなげます。

次のようにキーインしてください。

```
load "DM01/4.bas"
Ok
merge "DMmrg.asc"
Ok
█
```

さて、どうでしょうか。LISTコマンドで確かめてみてください。うまくマージできていましたか？

ここで1つだけ注意を。

先ほどマージ用のプログラムはアスキーセーブしましたが、もしこれを非アスキー形式でセーブしてマージしようとすると、

```
Sequential I/O only
```

というエラーになります。

アスキー形式はシーケンシャルにセーブするものです。そして上のエラーメッセージは「シーケンシャルしか受けつけられませんよ」という意味ですね。つまり、MERGEコマンドは、アスキー形式でセーブしたプログラムでないと通用しないのです。

プログラムの実行

それでは、このプログラムをＲＵＮさせて動作テストをしてみましょう。

［f・5］キーを押してください。すると、「アテナ　ノ　インサツヲシマス」という表示がでて、次にプリンタ有無の確認メッセージがその上に重ねて表示されてしまいます。

```
プリンタ ハ <y/n> ?マス
```

?マス ―― これが消えない

あとでＣＬＳ文を入れてなおしておきましょう。

このあとの実行方法はわかりますね。いろいろなコードをキーインしてテストしてみてください。私のほうは正常に走りました。

では、301 行をＣＬＳ文にし、305 行にＲＥＭ文を入れておきましょう。

```
301 CLS
305 '
```

そして、一度このプログラムをセーブしておきます。

```
save "DM01/4.bas"
```

通信文を印刷する

せっかく宛て名が表示できたのですから、コード入力による場合でプリンタをもっている方は、宛て名の次に通信文を印刷してみませんか？　簡単ですからやってみましょう。

いまセーブしたプログラムの 705 行から 730 行の宛て名印刷の次に通信文の印刷処理を入れればよいのです。この処理はサブルーチンにすることにして、次のステートメントを追加します。

```
733    GOSUB *LETTER
```

サブルーチンでは、まずその人の名前を印刷し、改行して通信文を印刷します。

次にプログラムリストを示しますが、サンプルとして、ナツメ社発行の書籍の読者向けの案内としました。ここはみなさんなりに、適当に変えてみてください。

```
6000 '
6010 *LETTER
6020 LPRINT NM$(ORDER2);"ｻﾏ"
6030 LPRINT " ｺﾉﾀﾋﾞ ﾂｷﾞﾉ ｼｮｾｷ ｶﾞ ﾊﾂﾊﾞｲ ﾆ ﾅﾘﾏｼﾀ"
6040 LPRINT "ﾉﾃﾞ ｵｼﾗｾ ｼﾏｽ."
6050 LPRINT
6060 LPRINT "  ｢PC-8001 ﾌﾟﾛｸﾞﾗﾐﾝｸﾞﾆｭｳﾓﾝ｣"
6070 LPRINT
6080 LPRINT " ﾓｸｼﾞ ｿﾉﾀ ﾉ ｼｮｳｻｲ ﾊ ﾄﾞｳﾌｳ ﾉ ﾊﾟﾝﾌﾚ"
6090 LPRINT "ﾄ ｦ ｺﾞﾗﾝｸﾀﾞｻｲ."
6100 LPRINT
6110 LPRINT "              ｶﾌﾞｼｷｶﾞｲｼｬ   ﾅﾂﾒ ｼｬ"
6120 FOR J=1 TO 10 : LPRINT : NEXT J
6130 RETURN
```

●漢字ＲＯＭを実装していれば、この通信文は漢字、カナまじりの文章として出力できます。

ごく普通のＬＰＲＩＮＴ文ですが、6120行のステートメントを入れ10行分の紙送りをしておきませんと、次の人の宛て名が続いて出力されてしまいます。

これを実行してみるとわかりますが、３人以上出力すると、２ページにまたがって印刷され、３人目の人の宛て名がミシン目の上にのってしまいます。これではちょっとまずいので、２人分印刷したら改ページするようにしておきます。

次のステートメントを追加してください。

```
731    RC=RC+1
734    IF RC=2 THEN RC=0 : LPRINT CHR$(12)
```

731行でレコードをカウントし、もし２人の処理がすんでいたら、734行で改ページ処理をしています。このときに、カウンタ用変数をゼロクリアしておくのを忘れないでください。

さて、これでプログラムをＲＵＮさせてみてください

実行例を右ページに示します。

この通信文はカタカナですので、少々読みずらいという欠点はありますが、宛て名と一緒に印刷できますので、紙送りのＬＰＲＩＮＴ文を調整することにより、宛て名を窓つき封筒の窓のところにくるようにし、パンフレットなどを同封すれば、十分実用的なプログラムとして活用できると思います。

```
111
aaaaaaaaaa
1-1-1
          AAAAAAA
                              サマ





AAAAAAA                      サマ
 コノタビ゛ ツキ゛ノ ショセキ ガ゛ ハツハ゛イ ニ ナリマシタ
ノテ゛ オシラセ シマス.

  「PC-8001 フ゜ログ゛ラミング゛ニュウモン」

 モクジ゛ ソノタ ノ ショウサイ ハ ト゛ウフウ ノ ハ゜ンフレ
ト ヲ ゴ゛ランクダ゛サイ.

                     カブ゛シキガ゛イシャ  ナツメ シャ
```

●上の文例で「パンフレット」の促音が抜けてしまいました。

DEF FN文を使う

DEF　FNはディファイン・ファンクションと読み、ユーザーによって作られた関数を定義するものです。

書式は次のとおりです。

DEF　FN〈関数名〉(〈パラメータリスト〉)＝〈関数の定義式〉

これをどこに使うかと申しますと、このプログラムの640行および645行でGETするセクタ番号とそのセクタ内でのレコード番号を決めていますが、この部分を「DEF　FN」に切り換えようというわけです。いまのままで、なんの支障もなく走るわけですから、わざわざこんなことをしなくてもよいのですが、この関数はかなり便利ですので、覚えておくとよいでしょう。

この関数をプログラム中で使うには、まず上の書式にならって関数

を定義しておかなければなりません。セクタ番号とレコード番号を求めるのですから、関数名はそれぞれSEC、RECとしておきます。そして、初期設定部分で次のようなステートメントを追加しておきます。

```
112 DEF FNSEC(X)=(X-1)¥3+2
114 DEF FNREC(X)=(X-1) MOD 3+1
```

これで関数を定義することができました。こうしておきますと、プログラムの中でいつでも呼び出すことができます。

では、640行と645行を次のように変更しますので、あらためて打ち込んでください。

```
640    SEC=FNSEC(ORDER1)
645    ORDER2=FNREC(ORDER1)
```

このように、カッコの中に必要なパラメータを入れますと、先ほど定義した数式にもとづいてSECおよびRECを計算してくれます。

カッコ内のパラメータは1次元とは限らず、2次元、3次元でもかまいませんので、複数の変数からなる算式を計算する場合など、とても便利です。

この節も少し長くなってしまいましたが、これでプログラムが完成しました。

いつものようにリナンバリングをして、プログラムの総リストをかかげておきます。

あ、そういう、このプログラムをもう一度セーブするのを忘れないでください。

```
renum 100,100
Ok
renum 1000,980
Ok
renum 2000,1140
Ok
renum 3000,2110
Ok
renum 4000,3080
Ok
renum 5000,4110
Ok
█
```

プログラムリスト

```
10 ' file name ==> DM01/4.bas
20 ' 1984.1.20
30 ' random data print out through code
40 ' save "DM01/4.bas"
50 '
100 CONSOLE 0,25,0,0 : WIDTH 80,25
110 DEFINT A-Z
120 DEF FNSEC(X)=(X-1)¥3+2
130 DEF FNREC(X)=(X-1) MOD 3+1
140 DIM NM$(3),PN$(3),AD$(3),BA$(3)
150 DIM SEX$(256)
160 '
170 CLS
180 GOSUB *OPNF01
190 GOSUB *OPNF02
200 GOSUB *READ.INDX : FLAG=1
210 DIM CODE(COUNT) : COUNT=0
220 GOSUB *READ.INDX : COUNT=0
230 '
240 LOCATE 25,4 : PRINT "ﾃﾞｰﾀ ﾉ ｲﾝｻﾂ ｦ ｼﾏｽ"
250 FOR I=1 TO 3000 : NEXT I
260 '
270 CLS
280 LOCATE 25,4 : PRINT "ﾌﾟﾘﾝﾀ ﾊ <y/n> ?"
290 Y$=INKEY$ : IF Y$="" THEN GOTO 290
300 IF Y$="Y" OR Y$="y" THEN P.FLAG=1
310 '
320 *LOOP
330 CLS
340 LOCATE 15,4 : INPUT "ｺｰﾄﾞ(END=999)";CD
350 IF CD=999 THEN GOTO *PRT
360 GOSUB *DAT.SEARCH
370 IF FOUND=-1 THEN GOTO *LOOP
380 ORDER1=FOUND
390 SEC=(ORDER1-1)¥3+2
400 ORDER2=(ORDER1-1) MOD 3+1
410 GET #1,SEC : GET #2,1
420 CLS
430 LOCATE 15,4 : PRINT NM$(ORDER2)
```

```
440 LOCATE 20,5 : PRINT "ｻﾝ ﾉ ｲﾝｻﾂ <y/n>?"
450 Y$=INKEY$ : IF Y$="" THEN GOTO 450
460 IF Y$="Y" OR Y$="y" THEN GOTO *TOROKU
470 IF Y$="N" OR Y$="n" THEN GOTO *LOOP
480 BEEP : BEEP :GOTO 450
490 '
500 *TOROKU
510 COUNT=COUNT+1 : CO(COUNT)=FOUND
520 GOTO *LOOP
530 '
540 *PRT
550 CLS
560 IF P.FLAG=1 THEN GOSUB *CHECK : GOTO 590
570 LOCATE 25,4 : PRINT "ｱﾃﾅ ｦ ﾋｮｳｼﾞｼﾏｽ"
580 FOR I=1 TO 5000 : NEXT I
590 FOR I=1 TO COUNT
600   ORDER1=CO(I)
610   SEC=FNSEC(ORDER1)
620   ORDER2=FNREC(ORDER1)
630   GET #1,SEC
640   IF P.FLAG=1 THEN GOTO 740
650   CLS
660   LOCATE 20,4 : PRINT NM$(ORDER2)
670   LOCATE 20,6 : PRINT PN$(ORDER2)
680   LOCATE 20,8 : PRINT AD$(ORDER2)
690   LOCATE 20,10: PRINT BA$(ORDER2)
700   LOCATE 25,12
710   PRINT "Hit RET key for next person!!"
720   Y$=INKEY$ : IF Y$="" THEN GOTO 720
730   GOTO 830
740   LPRINT PN$(ORDER2)
750   LPRINT AD$(ORDER2)
760   LPRINT BA$(ORDER2)
770   LPRINT TAB(5);NM$(ORDER2)
780   LPRINT TAB(15);"ｻﾏ"
790   FOR J=1 TO 5 : LPRINT : NEXT J
800   RC=RC+1
810   GOSUB *LETTER
820   IF RC=2 THEN RC=0 : LPRINT CHR$(12)
830 NEXT I
840 '
```

```
850 *FCLOSE
860 CLOSE #1,#2
870 '
880 CLS
890 LOCATE 25,4 : PRINT "ﾎｶ ﾉ ｼｺﾞﾄ ｦ ｼﾏｽｶ ?"
900 LOCATE 30,5 : PRINT "< y/n >"
910 Y$=INKEY$ : IF Y$="" THEN GOTO 910
920 IF Y$="Y" OR Y$="y" THEN RUN "DMMENU.bas"
930 IF Y$="N" OR Y$="n" THEN GOTO *FINISH
940 BEEP : BEEP :GOTO 890
950 '
960 *FINISH
970 END
1000 '
1010 *OPNF01
1020 OPEN "2:MAIL01.dat" AS #1
1030 FOR I=1 TO 3
1040   FIELD #1,85*(I-1) AS DUMMY1$,20 A
S NM$(I),6 AS PN$(I),39 AS AD$(I),20 AS
BA$(I)
1050 NEXT I
1060 RETURN
1070 '
1080 *OPNF02
1090 OPEN "2:MAIL01.dat" AS #2
1100 FOR I=1 TO 256
1110   FIELD #2,(I-1) AS DUMMY2$,1 AS SEX$(I)
1120 NEXT I
1130 RETURN
2000 '
2010 *READ.INDX
2020 OPEN "2:INDX01.dat" FOR INPUT AS #3
2030 WHILE NOT(EOF(3))
2040   COUNT=COUNT+1
2050   IF FLAG<>1 THEN GOTO 2070
2060   INPUT #3,CODE(COUNT) : GOTO 2080
2070   INPUT #3,DUMMY
2080 WEND
2090 CLOSE #3
2100 RETURN
3000 '
```

```
3010 *DAT.SEARCH
3020 FOUND=SEARCH(CODE,CD)
3030 IF FOUND<>-1 THEN RETURN
3040 LOCATE 20,6 : BEEP : BEEP
3050 PRINT "ﾄｳﾛｸ ｼﾃ ｱﾘﾏｾﾝ!!"
3060 FOR I=1 TO 3000 : NEXT I
3070 RETURN
4000 '
4010 *CHECK
4020 LOCATE 25,4 : PRINT "ﾌﾟﾘﾝﾀ ｦ ｾｯﾄｼﾃｸﾀﾞｻｲ"
4030 LOCATE 25,6 : PRINT "  Sure < y/n > ?"
4040 Y$=INKEY$ : IF Y$="" THEN GOTO 4040
4050 IF Y$="Y" OR Y$="y" THEN GOTO 4080
4060 IF Y$="N" OR Y$="n" THEN GOTO 4040
4070 BEEP : BEEP : GOTO 4040
4080 CLS
4090 LOCATE 25,4 : PRINT "ｱﾃﾅ ｦ ｲﾝｻﾂ ｼﾏｽ"
4100 RETURN
5000 '
5010 *LETTER
5020 LPRINT NM$(ORDER2);"ｻﾏ"
5030 LPRINT " ｺﾉﾀﾋﾞ ﾂｷﾞﾉ ｼｮｾｷ ｶﾞ ﾊﾂﾊﾞｲ ﾆ ﾅﾘﾏｼﾀ"
5040 LPRINT "ﾉﾃﾞ ｵｼﾗｾ ｼﾏｽ."
5050 LPRINT
5060 LPRINT "  ｢PC-8001 ﾌﾟﾛｸﾞﾗﾐﾝｸﾞﾆｭｳﾓﾝ｣"
5070 LPRINT
5080 LPRINT " ﾓｸｼﾞ ｿﾉﾀ ﾉ ｼｮｳｻｲ ﾊ ﾄﾞｳﾌｳ ﾉ ﾊﾟﾝﾌﾚ"
5090 LPRINT "ﾄ ｦ ｺﾞﾗﾝｸﾀﾞｻｲ."
5100 LPRINT
5110 LPRINT "                 ｶﾌﾞｼｷｶﾞｲｼｬ  ﾅﾂﾒ ｼｬ"
5120 FOR J=1 TO 10 : LPRINT : NEXT J
5130 RETURN
```

〈注〉1070行からのサブルーチンは2000行からとすればよかったのですが、ついうっかりしました。みなさんはリナンバリングしなおしてください。

1·9 メニュープログラム

メニュー画面とプログラムリストを示しますので、キーインし、セーブしてから、いろいろテストをしてみてください。

```
■|アテナ ガ゛キ メニュー|■

デ゛ータ ノ カキコミ・・・・・・1
デ゛ータ ノ シュウセイ・・・・・2
アテナ インサツ・・・・・・・・・3

    input number? ■
```

プログラムリスト

```
10 ' file name ==> DMMENU.bas
20 ' 1984.1.20
30 ' Menu program for direct mail No.1
40 ' save "DMMENU.bas"
50 '
100 CONSOLE 0,25,0,0 : WIDTH 80,25
110 DEFINT A-Z : CLS
120 '
130 LOCATE 25,4
140 PRINT "■|アテナ ガ゛キ メニュー|■"
150 LOCATE 25,6
160 PRINT "デ゛ータ ノ カキコミ・・・・・・1"
170 LOCATE 25,7
180 PRINT "デ゛ータ ノ シュウセイ・・・・・2"
190 LOCATE 25,8
200 PRINT "アテナ インサツ・・・・・・・・・3"
210 LOCATE 25,10
220 INPUT "    input number";NO
230 ON NO GOTO 250,260,270
240 '
250 RUN "DM01/1.bas"
260 RUN "DM01/2.bas"
270 RUN "DM01/3.bas"
```

第2章

もう少しきめ細かに

2・1　ファイルに、より資料性を

第１章のテーマ、「セクタのもう１つの使い方」いかがでしたか？実務面でデータをより有効に活用しようという点ではまだまだという感じがしますが、前著、

『ＰＣ―8801　ディスク・ファイル入門』

を読んでいただいた方々に、もう一歩進めてセクタの使い方を理解していただくことが目的でしたので、ある程度目的は達成できたのではないかと考えています。

また、プログラミングには前の本から引き続いて宛て名書きを選びましたが、ディスクの扱い方をマスターするためには、できたら１つのテーマを通して、いろいろなケースを考察したほうがよいと考えたからです。というのは、そのほうが、フィールド設定でも、プログラムの流れでも、あるいはランダムファイルのフォーマットの作り方でも、くり返し学ぶことになりますので、それだけ体得がはやいと思います。

よく英語などをマスターするのでも、いろいろな本を読むよりは、１、２冊の本を暗記するくらいに読めといわれます。ディスクの勉強でも原理はまったく同じだと思います。

そんなわけで、本章でもやはり宛て名書きのテーマにそくして、今度はもっと実用性をもたせるようにしたいと思います。

データファイルのフォーマットを決める

前節では、男女別、あるいは特定の人の宛て名を表示しましたが、この節ではもう少し範囲を広げて、職業別あるいは年齢制限などを加えてみます。

プログラムの流れ自体はそれほど変わりませんが、ランダムファイルのフォーマットが異なってきますので、前節のプログラムの一部を流用せず、作りなおしたほうがはやいようです。

ランダムファイルで業務管理をする場合、プログラミングの前にまず、どのような処理をしたいかを決め、それにもとづいてランダムファイルの形をしっかり作っておかなければなりません。

この節ではいったいどこまで達成できるかわかりませんが、処理してみたい項目を列挙し、私たちの当面の目的としておきましょう。

❶ **管理データを500人分にふやしたい。**

❷ **データ入力をし、修正もしたい。**

❸ **男女別、職業別、年齢別に宛て名を表示したい。**

❹ **たとえば毎年4月1日をベースにして、登録データの年齢を更改したい。**

あげればまだまだいろいろあると思いますが、だいたいこんなところを目ざして進みたいと思います。

この節で扱うランダムファイルディスクリプタは、

2:DM02/1.dat

とし、またインデックス用のシーケンシャルファイルのファイルディスクリプタは、

2:INDX02.dat

とします。前節のファイル名とは数字が1文字違うだけですから、間違わないように注意してください。

さて、ランダムファイルの設計ですが、次ページに概略を図示します。コメントを入れてありますので理解できると思いますが、簡単に説明しておきましょう。

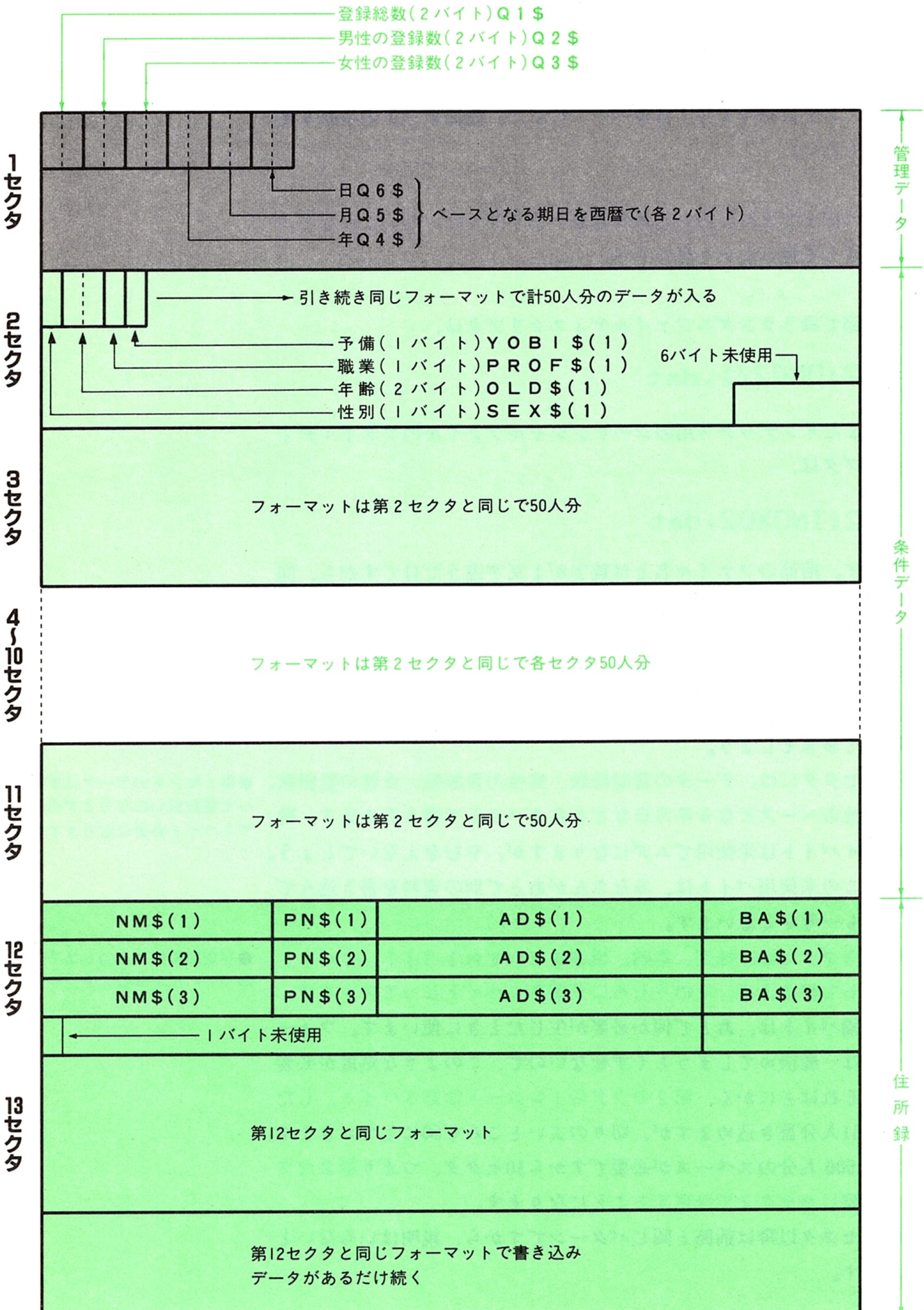
第1セクタには、データの登録総数、男性の登録数、女性の登録数、年齢更改のベースとなる年月日などを各2バイトで書き込みます。残りの244バイトは未使用でムダになりますが、やむをえないでしょう。また、この未使用バイトは、みなさんがあとで別の資料を書き込んでみるのも一考かと思います。

●第1セクタのデータはすべて整数扱いになりますので2バイト必要になります

第2セクタには、性別、年齢、職業をそれぞれ1バイト、2バイト、1バイトで書き込み、そのうしろに予備を1バイトとっておきます。この予備バイトは、あとで何か必要が生じたときに使います。フォーマットは一度決めてしまうとくずせないので、このような処置が必要です。それはとにかく、**第2セクタの1レコードは計5バイト**、したがって51人分書き込めますが、切りのよいところで50にしておきます。そして500人分のスペースが必要ですから10セクタ、つまり**第2セクタから第11セクタまで使用する**ことになります。

●年齢は整数で入力しますので2バイト必要です。

第12セクタ以降は前節と同じパターンですから、説明はいらないと思います。


登録総数(2バイト)Q1$
男性の登録数(2バイト)Q2$
女性の登録数(2バイト)Q3$
1セクタ
日Q6$
月Q5$
年Q4$
ベースとなる期日を西暦で(各2バイト)
管理データ
2セクタ
引き続き同じフォーマットで計50人分のデータが入る
予備(1バイト)YOBI$(1)
職業(1バイト)PROF$(1)
年齢(2バイト)OLD$(1)
性別(1バイト)SEX$(1)
6バイト未使用
3セクタ
フォーマットは第2セクタと同じで50人分
4〜10セクタ
フォーマットは第2セクタと同じで各セクタ50人分
条件データ
11セクタ
フォーマットは第2セクタと同じで50人分
12セクタ
NM$(1)
PN$(1)
AD$(1)
BA$(1)
NM$(2)
PN$(2)
AD$(2)
BA$(2)
NM$(3)
PN$(3)
AD$(3)
BA$(3)
1バイト未使用
13セクタ
第12セクタと同じフォーマット
第12セクタと同じフォーマットで書き込み
データがあるだけ続く
住所録

2・2　データ書き込みプログラム

それではこれからプログラムをキーインしていきますが、先ほども申し上げましたように、プログラムの流れは前章とだいたい同じですので、詳しい説明は省略します。

わからないところがあったら、もう一度前章を参照してください。

セクタ番号、レコード番号を決める

最初に、あるデータの、登録してある先頭データから数えて何番目かがわかったとき、これをもとにＧＥＴすべきセクタ番号およびそのセクタ内でのレコード番号を算出する式を考えてみましょう。

ここでは仮にＸ番目のデータとしておきます。

この方法については25〜27ページでくわしく説明しましたので、もしわからなかったら、その個所を読みなおしてください。

まず、Ｘ番目のデータが書き込まれるべき第２〜第11セクタ内でのセクタ番号は、次の式によって求められます。

セクタ番号１＝（Ｘ－１）￥50＋２

１セクタ内に50レコード書き込まれますから、Ｘから１を引いた数を50で割ると、その商はスタートラインとなるセクタ（この場合第２セクタ）から数えて何番目かを表わしていますから（スタートラインのセクタを０番目とする）、これにスタートラインのセクタ番号を加えておけばよいですね。

では、該当セクタ内でのレコード番号はどうなるでしょうか。

Ｘから１を引いた数を50で割るとその端数（余り）は、１つ前にずれていますので、これに１を加えておけばよいですね。したがって、次の算式で計算できます。

レコード番号１＝（Ｘ－１）ＭＯＤ　50＋１

次に、同じＸ番目のデータの第12セクタ以降での書き込みセクタ番号と、そのセクタ内でのレコード番号はどうなるでしょうか。

いまと同じように考えてみてください。

今度は第12セクタが基準（スタートライン）になりますから、セクタ番号は次のようになります。

セクタ番号２＝（Ｘ－１）￥３＋12

同様に、そのセクタ内でのレコード番号は、

レコード番号2＝（X－1）MOD 3＋1

となります。

これで初期設定の下準備ができました。前章で学んだDEF FN文を使って、セクタ番号、レコード番号を計算する関数を定義しておきましょう。

初期設定

この部分のプログラムを次に示します。

```
10 ' file name ==> DM02/1.bas
20 ' 1984.1.20
30 ' random data write
40 ' save "DM02/1.bas"
50 '
100 CONSOLE 0,25,0,0 : WIDTH 80,25
110 DEFINT A-Z
120 DEF FNSEC1(X)=(X-1)¥50+2
130 DEF FNREC1(X)=(X-1) MOD 50+1
140 DEF FNSEC2(X)=(X-1)¥3+12
150 DEF FNREC2(X)=(X-1) MOD 3+1
160 DIM NM$(3),PN$(3),AD$(3),BA$(3)
170 DIM SEX$(50),OLD$(50),PROF$(50),SPARE$(50)
```

この数を1にしないように!!

関数定義

プログラム名が前章の場合と似ていますので注意してください。

年齢、職業、予備のための配列変数をあらたに宣言しますが、1セクタに50人までの書き込みですから大きさは50でいいですね。

次は、ドライブ2へデータディスクをセットしたかどうかの確認ステートメントです。

```
180 '
190 CLS
200 LOCATE 25,4 : PRINT "ﾄﾞﾗｲﾌﾞ2ﾍ ﾃﾞｨｽｸｦﾄﾞｳｿﾞ"
210 LOCATE 25,6 : PRINT "    Sure <y/n> ?"
220 Y$=INKEY$ : IF Y$="" THEN GOTO 220
230 IF Y$="Y" OR Y$="y" THEN GOTO 250
                        ELSE GOTO 200
```

データディスクをドライブ２にセットしたら、次はファイルをオープンしますが、その前にファイルのＯＰＥＮ文とFIELD設定のサブルーチンを完成させておきましょう。

ファイルOPENとFIELD設定

この部分も、もうおわかりと思いますので、プログラムを示し、必要に応じて説明しておきます。

❶ 第２セクタから第11セクタまでのためのＦＩＥＬＤ設定

```
2000 '
2010 *OPNF01
2020 OPEN "2:MAIL02.dat" AS #1
2030 FOR I=1 TO 50
2040 FIELD #1,(I-1)*5 AS DUMMY1$,1 AS SEX$(I)
     ,2 AS OLD$(I),1 AS PROF$(I),1 AS SPARE$(I)
2050 NEXT I
2060 RETURN
```

ランダムファイルを１番バッファを使ってＯＰＥＮし、**性別、年齢、職業、予備のデータやりとりに使用する配列変数のフィールド幅を、ダミー変数を使って多重定義して設定しています。**

50人分ですから、50回ループさせればよいですね。そして１回ループするたびに**ダミー変数のフィールド幅は**５きざみにふくれあがり、**最終的には245バイトに設定**されます。

前にも述べましたが、ＧＥＴしてこのダミー変数を使わなければ何も問題は起こりません。

❷ 第12セクタ以降のためのＦＩＥＬＤ設定

```
3000 '
3010 *OPNF02
3020 OPEN "2:MAIL02.dat" AS #2
3030 FOR I=1 TO 3
3040   FIELD #2,85*(I-1) AS DUMMY2$,20 AS
NM$(I),6 AS PN$(I),39 AS AD$(I),20 AS BA$(I)
3050 NEXT I
3060 RETURN
```

このサブルーチンは前章とまったく同じです。

2番のバッファを使います。

OPEN文のファイル番号と、FIELD文のバッファ番号を同じにしておいてください。当然のことなのですが、よくこれを間違えたりして、正しいデータが得られないケースが多いようです。

❸ 第1セクタのデータやりとりのためのFIELD設定

```
4000 '
4010 *OPNF03
4020 OPEN "2:MAIL02.dat" AS #1
4030 FIELD #1,2 AS Q1$,2 AS Q2$,2 AS Q3$,
              2 AS Q4$,2 AS Q5$,2 AS Q6$
4040 RETURN
```

98ページの図を見ていただけばわかると思いますが、登録データ数、男性の登録数、女性の登録数、基準となる年、月、日用の変数に対して、それぞれ2バイト幅でフィールドを割り当てています。

これらのデータは整数で扱うから2バイトということですね。

使用するバッファは1番ですから、❶の場合と重複しますが、同時にファイルをOPENしないようにしますので支障はありません。

データ入力までの下準備

ランダムファイルオープンとフィールド設定のサブルーチンができましたので、次にデータ入力までの下準備をしておきます。

この部分でやるべきことは、いま作ったサブルーチンにジャンプさせてファイルをOPENするのはいうまでもありませんが、そのほかに、初回のデータ入力でなければ第1セクタに書き込まれている男性および女性の登録数をとりだしておかなければなりません（これはデータ入力のたびに加算し、終ったら第1セクタにふたたび書き込みます）。

また、初回の書き込みの場合には、インデックス用のシーケンシャルファイルをOUTPUTモードでOPEN、2回目以降ならAPPENDモードでOPENする必要がありますね。

そして最後に、ベースとなる年月日の入力です。

さあ、それではプログラムを次ページに示しますので、さっそくキーインしてみましょう。

```
240 '
250 CLS
260 GOSUB *OPNF03
270 IF LOF(1)=0 THEN CLOSE #1 : GOTO 320
280 GET #1,1        ←初回のデータ入力でなければ第1セクタをGET
290 M.COUNT=CVI(Q2$) ┐
300 F.COUNT=CVI(Q3$) ┘男女の登録数をカウンタに代入
310 CLOSE #1
320 GOSUB *OPNF01       第2～第11セクタをGETできるよ
330 GOSUB *OPNF02       うにあらかじめ第11セクタをPUT
340 IF LOF(1)=0 THEN PUT #1,11 : GOTO 380
350 GOSUB *READ.INDX : ORDER=COUNT
360 OPEN "2:INDX02.dat" FOR APPEND AS #3
370 GOTO 400
380 OPEN "2:INDX02.dat" FOR OUTPUT AS #3
390 '
400 LOCATE 25,3 : PRINT "デ゛ータ ヲ ニュウリョク シマス"
410 LOCATE 25,5 : PRINT "ヘ゛ース ノ 年月日 ヲ ト゛ウソ゛"
420 LOCATE 30,6 : INPUT "ナン年 <セイレキ>";YY
430 IF YY>1983 THEN GOTO 460
440 BEEP : BEEP : LOCATE 30,6
450 PRINT SPACE$(20) : GOTO 420
460 LOCATE 30,7 : INPUT "ナン月          ";MM
470 LOCATE 30,8 : INPUT "ナン日          ";DD
```

260行でランダムファイルをＯＰＥＮし、270行でファイル長をチェックし、もし０ならば初めての書き込みですからＣＬＯＳＥしてからジャンプさせます。

初回の書き込みではなければ、すでに男女の登録数が第１セクタにセットされているわけですから、これを取り出してカウンタ用変数に格納しておきます。280行から310行までがこの処理です。

320行、330行では、ランダムファイルをＯＰＥＮし、第２セクタ以降および第12セクタ以降にデータを書き込むためにフィールド設定をサブルーチンで行ないます。

340行のＰＵＮ文の意味がちょっとわかりにくいかと思いますので補足説明をしておきます。

ランダムファイルをＧＥＴするためには、あらかじめＰＵＴしてあることが必要条件です。しかしこれは、ＧＥＴするセクタをあらかじめＰＵＴするこいう意味では必ずしもなく、ＧＥＴしたいセクタよりもっと先のセクタをＰＵＴすれば、最初のセクタからこのセクタまで

●この本の校正時に気づいたのですが、340行のＰＵＴ文では第11セクタをＰＵＴしなくても第２セクタをＰＵＴしておけば十分なようです。というのは、２回目以降の書き込みの場合、少なくとも第12セクタはＰＵＴされているので、第３セクタ以降はいつでもＧＥＴできるからです。

は常にＧＥＴすることができます。

つまり、初回のデータ入力時に、一番終りの11セクタをＰＵＴしてしまえば、第２～第11セクタはいつでもＧＥＴできるのです。

350行からは、２回目以降のデータ入力の場合のデータカウント処理とシーケンシャルファイルのＯＰＥＮです（〈注〉このサブルーチンをここに示すのを忘れてしまいました。108ページにかかげますので入力してください）。

400行からは、ベースとなる年月日の入力です。この日付は、毎年一度、年齢の更新をしたいので、ある年度内では常に期首の年月日をキーインするようにします。たとえば、1984年４月１日を期首にしましたら、1985年３月31日までは1984年４月１日をベースとします。

430行は、うっかり昭和何年で入力しなかったかどうかのチェックです。

コードの入力部分

この部分は、前章とほぼ同じです。

次にステートメントを示します。

```
480 '
490 *LOOP
500 CLS
510 LOCATE 20,4 : INPUT "ｺｰﾄﾞ(END=999)";CD
520 IF CD=999 THEN GOTO *FCLOSE
530 LOCATE 20,5 : INPUT "ﾈﾝﾚｲ･･････････";A0
540 LOCATE 20,6 : INPUT "ｾｯｸｽ<m/f>････";A1$
550 GOSUB *PROFESSION
560 LOCATE 20,7 : INPUT "ﾅﾏｴ･･････････";A2$
570 LOCATE 20,8 : INPUT "ﾕｳﾋﾞﾝﾊﾞﾝｺﾞｳ･･";A3$
580 LOCATE 20,9 : INPUT "ｼﾞｭｳｼｮ･･･････";A4$
590 LOCATE 20,10: INPUT "ﾊﾞﾝﾁ･････････";A5$
600 LOCATE 25,12 : PRINT "SURE < y/n > ?"
610 Y$=INKEY$ : IF Y$="" THEN GOTO 610
620 IF Y$="N" OR Y$="n" THEN GOTO *LOOP
630 IF Y$="Y" OR Y$="y" THEN GOTO 640
    ELSE BEEP : BEEP :GOTO 600
640 IF A1$="m" OR A1$="M"
        THEN M.COUNT=M.COUNT+1
650 IF A1$="f" OR A1$="F"
        THEN F.COUNT=F.COUNT+1
```

530行で年齢を入力しています。

それから、550行のサブルーチンへのジャンプですが、ここでは職業を入力します。あとでサブルーチンリストを示しますが、入力方法は、いくつか職業を画面に表示させて番号を選択させるものです。

600行で、間違いないかどうかの確認をし、だいじょうぶでしたら、640行、650行で男女の登録数をカウントします。

職種メニューサブルーチン

550行でのジャンプ先のステートメントを次に示します。

```
7000 '
7010 *PROFESSION
7020 CLS
7030 LOCATE 25,4 : PRINT "ｼｮｸｷﾞｮｳ ﾊ ?"
7040 LOCATE 25,6 : PRINT "ｶｲｼｬ ﾔｸｲﾝ·····1"
7050 LOCATE 25,7 : PRINT "ｻﾗﾘｰﾏﾝ········2"
7060 LOCATE 25,8 : PRINT "ｶﾞｸｾｲ·········3"
7070 LOCATE 25,9 : PRINT "ｼｭﾌ···········4"
7080 LOCATE 25,10: PRINT "ｿﾉﾀ···········5"
7090 LOCATE 25,12: INPUT "     input No.";PF$
7100 CLS
7110 LOCATE 20,3 : PRINT "ｺｰﾄﾞ(END=999)";CD
7120 LOCATE 20,4 : PRINT "ﾈﾝﾚｲ·········";A0
7130 LOCATE 20,5 : PRINT "ｾｯｸｽ<m/f>····";A1$
7140 LOCATE 20,6 : PRINT "ｼｮｸｷﾞｮｳ······";PF$
7150 RETURN
```

ごらんのように、1から5までの職業を表示し、番号を選択させます。

職業については、この本が終ってから、みなさんにとって役立つように分類してみてください。

7090行のINPUT文で番号を入力しますと、一度画面をクリアして、前ページのリスト560行の氏名入力につながるように、コード、年齢、性別、職業を表示してからリターンさせます。

データの書き込み部分

1名分のデータ入力が終りましたら、次はランダムファイルへのデータ書き込みです。

プログラムは次ページのとおりです。

```
660 '
670 *DAT.WRITE1
680 ORDER=ORDER+1
690 IF ORDER>500 THEN GOTO *ACCIDENT
700 SEC1=FNSEC1(ORDER)
710 REC1=FNREC1(ORDER)
720 SEC2=FNSEC2(ORDER)
730 REC2=FNREC2(ORDER)
740 IF REC2=1 THEN GOTO 760
750 GET #2,SEC2
760 GET #1,SEC1
770 WRITE #3,CD
780 IF REC2<>1 THEN GOTO 850
790 FOR I=2 TO 3
800 LSET NM$(I)=""
810 LSET PN$(I)=""
820 LSET AD$(I)=""
830 LSET BA$(I)=""
840 NEXT I
850 LSET SEX$(REC1) =A1$
860 LSET OLD$(REC1) =MKI$(A0)
870 LSET PROF$(REC1)=PF$
880 LSET NM$(REC2)  =A2$
890 LSET PN$(REC2)  =A3$
900 LSET AD$(REC2)  =A4$
910 LSET BA$(REC2)  =A5$
920 PUT #1,SEC1 : PUT #2,SEC2
930 GOTO *LOOP
```

700〜730行：書き込みセクタとそのセクタ内でのレコード番号を計算

740行：第12セクタ以降でレコード番号が1のときは次の行でGETできないのでジャンプ

800〜830行：前のデータをクリア

850〜870行：1番バッファを使用

880〜910行：2番バッファを使用

690行はデータ数が制限オーバーした場合の処理でしたね。

700行から730行で、初期設定で定義した関数を使って、書き込みセクタとそのセクタ内でのレコード番号を算出しています。

740行では、第12セクタ以降への書き込みで、かつレコード番号が1のときの処理をしています。このステートメントがないと、この場合次のGET文で

`Input past end in 750`

のエラーになります。

850行から870行までは1番バッファでＬＳＥＴ、880行から910行までは2番バッファでＬＳＥＴしています。そして、このセットされたデータを920行でＰＵＴしているわけです。

次に、690 行での飛び先のステートメントを示しておきます。これは、前章の場合とまったく同じです。

```
6000 '
6010 *ACCIDENT
6020 CLOSE #1,#2,#3
6030 CLS : BEEP : BEEP
6040 LOCATE 25,4 : PRINT "データ カキコミ デキマセン!"
6050 FOR I=1 TO 300 : NEXT I
6060 GOTO 1080
```

510行でコードに「999」が入力された場合には、データ入力は終りですから、ファイルをCLOSEし、第1セクタへのデータ書き込みを行ないます。

プログラムは次のとおりです。

```
940 '
950 *FCLOSE
960 CLOSE #1,#2,#3
970 '
980 *DAT.WRITE2
990 GOSUB *OPNF03
1000 LSET Q1$=MKI$(ORDER)
1010 LSET Q2$=MKI$(M.COUNT)
1020 LSET Q3$=MKI$(F.COUNT)
1030 LSET Q4$=MKI$(YY)
1040 LSET Q5$=MKI$(MM)
1050 LSET Q6$=MKI$(DD)
1060 PUT #1,1 : CLOSE #1
```

一度、いままで使用していたファイルをすべてCLOSEしてしまいます。

そしてあらためてサブルーチンへ飛ばしてファイルをOPENし、バッファ上でLSETをして第1セクタに書き込みます。

98ページの図を見ればわかるように、第1セクタには総登録データ数、男性と女性のデータ数、年月日ですね。

すべて整数データですから、MKI$を使ってストリングに変換してからLSETをします。

プログラムのエンド部分

最後にエンド処理のプログラムをキーインしておきましょう。

```
1070 '
1080 CLS
1090 LOCATE 25,4 : PRINT "ﾎｶ ﾉ ｼｺﾞﾄ ｦ ｼﾏｽｶ ?"
1100 LOCATE 30,5 : PRINT "< y/n >"
1110 Y$=INKEY$ : IF Y$="" THEN GOTO 1110
1120 IF Y$="Y" OR Y$="y" THEN RUN "DMmenu.bas"
1130 IF Y$="N" OR Y$="n" THEN GOTO *FINISH
1140 BEEP : BEEP :GOTO 1090
1150 '
1160 *FINISH
1170 END
```

それではこのプログラムをセーブしておきましょう。

```
save "DM02/1.bas"
```

次に示すのは、104ページで入力するはずのサブルーチンです。

ついうっかりしていました。おわびいたします。編集長にだいぶしかられそうです。

```
5000 '
5010 *READ.INDX
5020 OPEN "2:INDX02.dat" FOR INPUT AS #3
5030 WHILE NOT(EOF(3))
5040   COUNT=COUNT+1
5050   INPUT #3,DUMMY
5060 WEND
5070 CLOSE #3
5080 RETURN
```

2・3　プログラムの動作テスト

プログラムはセーブしましたか？　忘れないでください。

さあ、それではプログラムをRUNさせてデータを入力してみましょう。入力用のサンプルデータを用意しましたので次に示します。

コード	年齢	性別	職業	氏　名	郵便番号	住　所	番地
101	42	m	2	AAAAA	111	aaaaaaaaaa	1-1
201	10	f	3	BBBBB	222	bbbbbbbbbb	2-1
102	50	m	1	CCCCC	333	cccccccccc	1-2
202	27	f	3	DDDDD	444	dddddddddd	2-2
103	30	m	4	EEEEE	555	eeeeeeeeee	1-3
203	19	f	3	FFFFF	666	ffffffffff	2-3
104	32	m	2	GGGGG	777	gggggggggg	1-4

またまた、ずいぶんそっけないデータですが、正常に動くかどうか確認するだけですので、入力しやすいほうがよいでしょう。

また、職業の欄には数字が入れてありますが、これは画面に職種のメニューが表示されますから、そのときキーインする番号です。

それでは f・5 キーを押してRUNさせてください。

最初にドライブ2へディスクを入れたかどうかの確認メッセージが現れ、次にベースの年月日の入力となります。ここでは1984年4月1日としておきます。

そして、上のデータの入力です。一気にやってしまいましょう。

7つのデータの入力が終りましたら、いまのプログラムの不要行を次のようにデリートします。

```
delete 10-150
Ok
delete 180-1170
Ok
delete 4040-7150
Ok
2060
3060
█
```

これはどういう意味だかわかりますか？　第1章を思い出してください。

そうですね、ファイルオープンとフィールド設定のところだけ残しておくのです。そしてこれをRUNさせて、ダイレクトモードでデータ書き込みの確認をしようというわけです。

RUNする前にもう2か所、4020行と4030行のバッファ番号を、次に緑色で示すように3番に変更しておいてください。

```
4020 OPEN "2:MAIL02.dat" AS #3
4030 FIELD #3,2 AS Q1$,2 AS Q2$,2 AS Q3$,
              2 AS Q4$,2 AS Q5$,2 AS Q6$
```

なぜだかわかりますね。すでに2020行でバッファ1は使っているからです。なお、このファイルオープンのプログラムは次のようにセーブしておいてください。次節で使います。

```
save "DMTEST" ↵
```

さあ、それでは実行してください。「OK」が表示されたら、まず第2セクタの書き込み状況を調べてみましょう。

ダイレクトモードで次のようにキーインしてください。

```
get #1,2
Ok
for i=1 to 7:? sex$(i);cvi(old$(i));prof$(i):next
m 42 2
f 10 3
m 50 1
f 27 3
m 30 4
f 19 3
m 32 2
Ok
```

第2セクタをGET

性別、年齢、職業の順に7人のデータが書き込まれている

どうですか？　間違いなく7人のデータが書き込まれていましたね。

では次に第13セクタをチェックしてみましょう。名前とか住所は12セクタから1セクタ3人あて書き込まれていますから、第13セクタというと、4人目から6人目までの人のデータということになります。

使用バッファは2番です。

次のようにキーインしてください。

```
get #2,13
Ok
for i=1 to 3:? nm$(i):? pn$(i):? ad$(i):? ba$(i):next
DDDDD
444          ──4人目のデータ
dddddddddd
2-2
EEEEE
555          ──5人目のデータ
eeeeeeeeee
1-3
FFFFF
666          ──6人目のデータ
ffffffffff
2-3
Ok
■
```

これも異常なさそうです。

それでは第1セクタはどうでしょうか。

今度は3番バッファを使ってGETしてください。データは数値データでしたから、CVIで変換して表示します。

```
get #3,1
Ok
? cvi(q1$)cvi(q2$)cvi(q3$)cvi(q4$)cvi(q5$)cvi(q6$)
 7  4  3  1984  4  1
Ok
■
```

全データ数が7、うちわけは男性4、女性3、ベースの日付が1984年4月1日、これもうまくいっていました。

最後に必ずCLOSEをしておいてください。

なお、正しい書き込みデータが得られなかった方は、次にプログラムリストを示しますから、もう一度チェックし、ランダムファイルとシーケンシャルファイルをKILLコマンドで消してから、再度データ入力からやってみてください。

```
10 ' file name ==> DM02/1.bas
20 ' 1984.1.20
30 ' random data write
40 ' save "DM02/1.bas"
50 '
100 CONSOLE 0,25,0,0 : WIDTH 80,25
110 DEFINT A-Z
120 DEF FNSEC1(X)=(X-1)¥50+2
130 DEF FNREC1(X)=(X-1) MOD 50+1
140 DEF FNSEC2(X)=(X-1)¥3+12
150 DEF FNREC2(X)=(X-1) MOD 3+1
160 DIM NM$(3),PN$(3),AD$(3),BA$(3)
170 DIM SEX$(50),OLD$(50),PROF$(50),SPARE$(50)
180 '
190 CLS
200 LOCATE 25,4 : PRINT "ﾄﾞﾗｲﾌﾞ2ﾍ ﾃﾞｨｽｸｦﾄﾞｳｿﾞ"
210 LOCATE 25,6 : PRINT "     Sure <y/n> ?"
220 Y$=INKEY$ : IF Y$="" THEN GOTO 220
230 IF Y$="Y" OR Y$="y" THEN GOTO 250
                         ELSE GOTO 200
240 '
250 CLS
260 GOSUB *OPNF03
270 IF LOF(1)=0 THEN CLOSE #1 : GOTO 320
280 GET #1,1
290 M.COUNT=CVI(Q2$)
300 F.COUNT=CVI(Q3$)
310 CLOSE #1
320 GOSUB *OPNF01
330 GOSUB *OPNF02
340 IF LOF(1)=0 THEN PUT #1,11 : GOTO 380
350 GOSUB *READ.INDX : ORDER=COUNT
360 OPEN "2:INDX02.dat" FOR APPEND AS #3
370 GOTO 400
380 OPEN "2:INDX02.dat" FOR OUTPUT AS #3
390 '
400 LOCATE 25,3 : PRINT "ﾃﾞｰﾀ ｦ ﾆｭｳﾘｮｸ ｼﾏｽ"
410 LOCATE 25,5 : PRINT "ﾍﾞｰｽ ﾉ 年月日 ｦ ﾄﾞｳｿﾞ"
420 LOCATE 30,6 : INPUT "ﾅﾝ年 <ｾｲﾚｷ>";YY
430 IF YY>1983 THEN GOTO 460
440 BEEP : BEEP : LOCATE 30,6
450 PRINT SPACE$(20) : GOTO 420
460 LOCATE 30,7 : INPUT "ﾅﾝ月         ";MM
```

```
470 LOCATE 30,8 : INPUT "ナン日        ";DD
480 '
490 *LOOP
500 CLS
510 LOCATE 20,4 : INPUT "コード(END=999)";CD
520 IF CD=999 THEN GOTO *FCLOSE
530 LOCATE 20,5 : INPUT "ネンレイ・・・・・・・・・";A0
540 LOCATE 20,6 : INPUT "セックス<m/f>・・・・";A1$
550 GOSUB *PROFESSION
560 LOCATE 20,7 : INPUT "ナマエ・・・・・・・・・・";A2$
570 LOCATE 20,8 : INPUT "ユウビンバンゴウ・・";A3$
580 LOCATE 20,9 : INPUT "ジュウショ・・・・・・・";A4$
590 LOCATE 20,10: INPUT "バンチ・・・・・・・・・";A5$
600 LOCATE 25,12 : PRINT "SURE < y/n > ?"
610 Y$=INKEY$ : IF Y$="" THEN GOTO 610
620 IF Y$="N" OR Y$="n" THEN GOTO *LOOP
630 IF Y$="Y" OR Y$="y" THEN GOTO 640
    ELSE BEEP : BEEP :GOTO 600
640 IF A1$="m" OR A1$="M"
       THEN M.COUNT=M.COUNT+1
650 IF A1$="f" OR A1$="F"
       THEN F.COUNT=F.COUNT+1
660 '
670 *DAT.WRITE1
680 ORDER=ORDER+1
690 IF ORDER>500 THEN GOTO *ACCIDENT
700 SEC1=FNSEC1(ORDER)
710 REC1=FNREC1(ORDER)
720 SEC2=FNSEC2(ORDER)
730 REC2=FNREC2(ORDER)
740 IF REC2=1 THEN GOTO 760
750 GET #2,SEC2
760 GET #1,SEC1
770 WRITE #3,CD
780 IF REC2<>1 THEN GOTO 850
790 FOR I=2 TO 3
800 LSET NM$(I)=""
810 LSET PN$(I)=""
820 LSET AD$(I)=""
830 LSET BA$(I)=""
840 NEXT I
850 LSET SEX$(REC1) =A1$
860 LSET OLD$(REC1) =MKI$(A0)
```

```
870 LSET PROF$(REC1)=PF$
880 LSET NM$(REC2)  =A2$
890 LSET PN$(REC2)  =A3$
900 LSET AD$(REC2)  =A4$
910 LSET BA$(REC2)  =A5$
920 PUT #1,SEC1 : PUT #2,SEC2
930 GOTO *LOOP
940 '
950 *FCLOSE
960 CLOSE #1,#2,#3
970 '
980 *DAT.WRITE2
990 GOSUB *OPNF03
1000 LSET Q1$=MKI$(ORDER)
1010 LSET Q2$=MKI$(M.COUNT)
1020 LSET Q3$=MKI$(F.COUNT)
1030 LSET Q4$=MKI$(YY)
1040 LSET Q5$=MKI$(MM)
1050 LSET Q6$=MKI$(DD)
1060 PUT #1,1 : CLOSE #1
1070 '
1080 CLS
1090 LOCATE 25,4 : PRINT "ﾎｶ ﾉ ｼｺﾞﾄ ｦ ｼﾏｽｶ ?"
1100 LOCATE 30,5 : PRINT "< y/n >"
1110 Y$=INKEY$ : IF Y$="" THEN GOTO 1110
1120 IF Y$="Y" OR Y$="y" THEN RUN "DMmenu.bas"
1130 IF Y$="N" OR Y$="n" THEN GOTO *FINISH
1140 BEEP : BEEP :GOTO 1090
1150 '
1160 *FINISH
1170 END
2000 '
2010 *OPNF01
2020 OPEN "2:MAIL02.dat" AS #1
2030 FOR I=1 TO 50
2040 FIELD #1,(I-1)*5 AS DUMMY1$,1 AS SEX$(I)
     ,2 AS OLD$(I),1 AS PROF$(I),1 AS SPARE$(I)
2050 NEXT I
2060 RETURN
3000 '
3010 *OPNF02
3020 OPEN "2:MAIL02.dat" AS #2
3030 FOR I=1 TO 3
```

```
3040   FIELD #2,85*(I-1) AS DUMMY2$,20 AS
NM$(I),6 AS PN$(I),39 AS AD$(I),20 AS BA$(I)
3050 NEXT I
3060 RETURN
4000 '
4010 *OPNF03
4020 OPEN "2:MAIL02.dat" AS #1
4030 FIELD #1,2 AS Q1$,2 AS Q2$,2 AS Q3$,
              2 AS Q4$,2 AS Q5$,2 AS Q6$
4040 RETURN
5000 '
5010 *READ.INDX
5020 OPEN "2:INDX02.dat" FOR INPUT AS #3
5030 WHILE NOT(EOF(3))
5040   COUNT=COUNT+1
5050   INPUT #3,DUMMY
5060 WEND
5070 CLOSE #3
5080 RETURN
6000 '
6010 *ACCIDENT
6020 CLOSE #1,#2,#3
6030 CLS : BEEP : BEEP
6040 LOCATE 25,4 : PRINT "ﾃﾞｰﾀ ｶｷｺﾐ ﾃﾞｷﾏｾﾝ!"
6050 FOR I=1 TO 300 : NEXT I
6060 GOTO 1080
7000 '
7010 *PROFESSION
7020 CLS
7030 LOCATE 25,4 : PRINT "ｼｮｸｷﾞｮｳ ﾊ ?"
7040 LOCATE 25,6 : PRINT "ｶｲｼｬ ﾔｸｲﾝ･････1"
7050 LOCATE 25,7 : PRINT "ｻﾗﾘｰﾏﾝ････････2"
7060 LOCATE 25,8 : PRINT "ｶﾞｸｾｲ･････････3"
7070 LOCATE 25,9 : PRINT "ｼｭﾌ･･････････4"
7080 LOCATE 25,10: PRINT "ｿﾉﾀ･･････････5"
7090 LOCATE 25,12: INPUT "     input No.";PF$
7100 CLS
7110 LOCATE 20,3 : PRINT "ｺｰﾄﾞ(END=999)";CD
7120 LOCATE 20,4 : PRINT "ﾈﾝﾚｲ･････････";A0
7130 LOCATE 20,5 : PRINT "ｾｯｸｽ<m/f>････";A1$
7140 LOCATE 20,6 : PRINT "ｼｮｸｷﾞｮｳ･･････";PF$
7150 RETURN
```

2·4 データ修正のプログラム

この節では、住所変更、あるいは改姓などの場合のデータ修正をテーマとしていますが、住所録の不要になった人はキャンセルし、そのコード番号を新規の人に割り当て、性別、年齢、職業などをこのプログラムで入力すれば、幅広く利用することができます。

とくに、このプログラムでは書き込み人数を500人に限定していますので、500人を超えるような場合には、無効の人のデータの上に新規の人のそれを書き込んでいけば、効率よく使えると思います。

書き込みプログラムの一部を流用

それでは始めますが、前節のプログラムの3分の1程度がそのまま流用できますので、不要行を削除して使うことにしましょう。

まず、前節のプログラムをロードしてください。

```
load "DM02/1.bas"
```

ロードできましたら、次のようにダイレクトモードで不要行を削除します。

```
delete 240-1060
Ok
delete 6000-7150
Ok
■
```

240行から1060まではメイン部分になりますので、全面的に作りなおします。6000行から6060行までは不要ですし、7000行からのサブルーチンは作りなおします。

それでは最初に、先頭のREM文をスクリーンエディタで修正しておきましょう。

```
10 ' file name ==> DM02/2.bas
20 ' 1984.1.20
30 ' random data correct
40 ' save "DM02/2.bas"
50 '
```

ファイルオープンと配列宣言

次は、データ修正はコード入力によりますので、シーケンシャルファイルをOPENしてコード用配列を宣言します。そして、ランダムファイルをOPENして、メインルーチンへの下準備をします。

プログラムを次に示します。

```
240 '
250 CLS
260 GOSUB *READ.INDX : FLAG=1
270 DIM CO(COUNT) : COUNT=0
280 GOSUB *READ.INDX           ←── 配列にコードを読み込む
290 CLOSE #3
300 GOSUB *OPNF03
310 GET #1,1
320 M.COUNT=CVI(Q2$) ┐
330 F.COUNT=CVI(Q3$) ┘──────── 男女数をカウンタに格納
340 CLOSE #1
350 GOSUB *OPNF01
360 GOSUB *OPNF02
370 '
380 LOCATE 25,3 : PRINT "デ゛ータ ノ シュウセイ ヲシマス"
390 FOR I=1 TO 3000 : NEXT I
```

このプログラムでは既存のデータをキャンセルして新規の人に差し換えることも想定していますので、その場合当然、男性と女性の数は変動しますから、現時点での男女数をあらかじめカウンタに入れておきます。300行から340行までがその処理です。

350行、360行でランダムファイルOPENとFIELD設定のサブルーチンへ飛ばしています。

260行の1回目のGOSUB文ではシーケンシャルファイルの読み飛ばし、280行の2回目の場合は配列へのコード読み込みですから、サブルーチンに次の2行のステートメントを追加しなければなりませんね。

これは前に勉強しましたからわかりますね。

```
5042   IF FLAG<>1 THEN GOTO 5050
5044   INPUT #3,CO(COUNT) : GOTO 5060
```

コードの入力

次はコードの入力と、データの確認です。

```
400 '
410 *LOOP
420 CLS
430 LOCATE 20,4 : INPUT "ｺｰﾄﾞ<END=999>";CD
440 IF CD=999 THEN GOTO *FCLOSE
450 GOSUB *DAT.SEARCH
460 IF FOUND=-1 THEN GOTO *LOOP
470 SEC1=FNSEC1(FOUND)
480 REC1=FNREC1(FOUND)
490 SEC2=FNSEC2(FOUND)
500 REC2=FNREC2(FOUND)
510 GET #1,SEC1 : GET #2,SEC2
520 '
530 CLS
540 LOCATE 20,4 : PRINT NM$(REC2)
550 LOCATE 25,5 : PRINT "ｻﾝ ﾉ ｼｭｳｾｲ <y/n>?"
560 Y$=INKEY$ : IF Y$="" THEN GOTO 560
570 IF Y$="Y" OR Y$="y" THEN GOTO *CORRECT
580 IF Y$="N" OR Y$="n" THEN GOTO *LOOP
590 BEEP : BEEP :GOTO 560
```

コードが入力されると、サブルーチンへジャンプし、登録してあるかどうかチェックします。もし、あれば、470行から500行でそのデータの書き込まれているセクタ番号、セクタ内でのレコード番号を計算します。そして510行でGETです。

次にデータサーチのサブルーチンをキーインしましょう。

```
6000 '
6010 *DAT.SEARCH
6020 FOUND=SEARCH(CO,CD)
6030 IF FOUND<>-1 THEN RETURN
6040 LOCATE 20,6 : BEEP : BEEP
6050 PRINT "ﾄｳﾛｸ ｼﾃ ｱﾘﾏｾﾝ!!"
6060 FOR I=1 TO 2000 : NEXT I
6070 RETURN
```

これも前にやりましたね。

データの修正部分

プログラムは次のとおりです。

```
600 '
605 *CORRECT
610 CLS
615 LOCATE 20,2 : PRINT "ｼｭｳｾｲﾅｹﾚﾊﾞ RET"
620 LOCATE 20,4
625 PRINT "ﾈﾝﾚｲ··········";CVI(OLD$(REC1))
630 LOCATE 20,5 : INPUT "    ==>";A0
635 B0=CVI(OLD$(REC1))
640 IF A0=0 THEN A0=CVI(OLD$(REC1))
645 LOCATE 20,6
650 PRINT "ｾｯｸｽ<m/f>····";SEX$(REC1)
655 LOCATE 20,7 : INPUT "    ==>";A1$
660 B1$=SEX$(REC1)
665 IF A1$="" THEN A1$=SEX$(REC1) ELSE IF
    A1$="m" OR A1$="M" THEN M.FLAG=1
    ELSE F.FLAG=1
670 GOSUB *PROFESSION
675 LOCATE 20,10
680 PRINT "ﾅﾏｴ···········";NM$(REC2)
685 LOCATE 20,11 : INPUT "    ==>";A2$
690 IF A2$="" THEN A2$=NM$(REC2)
695 LOCATE 20,12
700 PRINT "ﾕｳﾋﾞﾝﾊﾞﾝｺﾞｳ··";PN$(REC2)
705 LOCATE 20,13 : INPUT "    ==>";A3$
710 IF A3$="" THEN A3$=PN$(REC2)
715 LOCATE 20,14
720 PRINT "ｼﾞｭｳｼｮ········";AD$(REC2)
725 LOCATE 20,15 : INPUT "    ==>";A4$
730 IF A4$="" THEN A4$=AD$(REC2)
735 LOCATE 20,16
740 PRINT "ﾊﾞﾝﾁ··········";BA$(REC2)
745 LOCATE 20,17 : INPUT "    ==>";A5$
750 IF A5$="" THEN A5$=BA$(REC2)
755 LOCATE 25,19 : PRINT "Sure < y/n >?"
760 Y$=INKEY$ : IF Y$="" THEN GOTO 760
765 IF Y$="Y" OR Y$="y" THEN *DAT.WRITE1
770 IF Y$="N" OR Y$="n" THEN *LOOP
775 BEEP : BEEP :GOTO 760
```

どの項目も、修正がなければリターンキーを押すという点は前章と同じです。

行番号を５きざみにしましたが、10きざみにすると、1070行までにおさまりそうもないので、このようにしました。

635行、660行で、Ｂ０およびＢ１＄に修正前のデータを代入していますが、これは、670行で職業修正のサブルーチンへジャンプすると画面がクリアされますので、飛び先から帰ってきたときに、ふたたびいまのような画面を再現させるために古いデータを保存しておくのです。

665行は、新規のデータを入力したりして性別の変更があった場合に、男女別のフラグを立てておく処理です。このフラグいかんによって、あとで男女数の増減をします。

このステートメントの意味は、「もしリターンキーが押されていれば古いデータをそのまま生かすことになりＥＬＳＥ以下は実行しません。もしリターンキーでなかったらＭかＦが入力されたことになりますから、対応したフラグを立てる」ということになります。

次は職業修正のサブルーチンです。

```
7000 '
7010 *PROFESSION
7020 CLS
7030 LOCATE 25,3
7040 PRINT "ｼｮｸｷﾞｮｳ･･････";PROF$(REC1)
7050 LOCATE 25,6 : PRINT "ｶｲｼｬ ﾔｸｲﾝ･････1"
7060 LOCATE 25,7 : PRINT "ｻﾗﾘｰﾏﾝ････････2"
7070 LOCATE 25,8 : PRINT "ｶﾞｸｾｲ･････････3"
7080 LOCATE 25,9 : PRINT "ｼｭﾌ･･･････････4"
7090 LOCATE 25,10: PRINT "ｿﾉﾀ･･･････････5"
7100 LOCATE 25,4 : INPUT "    ==>";PF$
7110 B2$=PROF$(REC1)
7120 IF PF$="" THEN PF$=PROF$(REC1)
7130 CLS
7140 LOCATE 20,2 : PRINT "ｼｭｳｾｲﾅｹﾚﾊﾞ RET"
7150 LOCATE 20,4 : PRINT "ﾈﾝﾚｲ･････････";B0
7160 LOCATE 20,5 : PRINT "    ==>"A0
7170 LOCATE 20,6 : PRINT "ｾｯｸｽ<m/f>････";B1$
7180 LOCATE 20,7 : PRINT "    ==> ";A1$
7190 LOCATE 20,8 : PRINT "ｼｮｸｷﾞｮｳ･･････";B2$
7200 LOCATE 20,9 : PRINT "    ==> ";PF$
7210 RETURN
```

これは、職業メニューを表示し、修正がなければリターンキー、あれば番号を入力するようにしてあります。

複雑ではありませんので、プログラムを読んでいただけばわかると思います。

データの書き込み

プログラムは次のとおりです。

119ページの755行の確認ステートメントで、OKならデータの書き込みとなります。

```
780 '
785 *DAT.WRITE1
790 LSET SEX$(REC1) =A1$
795 LSET OLD$(REC1) =MKI$(A0)
800 LSET PROF$(REC1)=PF$
805 LSET NM$(REC2)  =A2$
810 LSET PN$(REC2)  =A3$
815 LSET AD$(REC2)  =A4$
820 LSET BA$(REC2)  =A5$
825 PUT #1,SEC1 : PUT #2,SEC2
830 IF M.FLAG=1 THEN M.COUNT=M.COUNT+1
: F.COUNT=F.COUNT-1 : M.FLAG=0 : GOTO 840
835 IF F.FLAG=1 THEN F.COUNT=F.COUNT+1
: M.COUNT=M.COUNT-1 : F.FLAG=0
840 A0=0 : PF$="" : A1$=""
845 A2$="" : A3$="" : A4$="" : A5$=""
850 GOTO *LOOP
```

変わったところというと、830行、835行で男女数の変動があったときの調整を行なっています。その判断規準は、先ほど立てておいたフラグですね。

このプログラムはデータ追加ではありませんから、データ総数は変わりません。ですから、たとえば男性が1人ふえれば女性は1人減らしておかなければなりませんね。

850行でコード入力へジャンプし、もし「999」が入力されれば、ファイルをいったんCLOSEし、第1セクタに資料を書き込みます。

プログラムを次ページに示します。

```
855 '
860 *FCLOSE
865 CLOSE #1,#2,#3
870 '
875 *DAT.WRITE2
880 GOSUB *OPNF03
885 GET #1,1
890 LSET Q2$=MKI$(M.COUNT)
895 LSET Q3$=MKI$(F.COUNT)
900 PUT #1,1 : CLOSE #1
```

最後にファイルをCLOSEするのを忘れないでください。

これで、データ修正プログラムは完成です。
まず忘れないうちにセーブしておきましょう。

```
save "DM02/2.bas"
```

プログラムの実行

さて、それではRUNさせてみましょう。正常に走るかどうかチェックするデータとして、109ページの7番目の人を新規の人にすることにして次のようにしてみます。コードは変わりませんよ。

コード	**104**	氏名	**ABCDE**
年齢	**21**	郵便番号	**123**
性別	**f**	住所	**abcdefghij**
職業	**3**	番地	**1-2-3**

準備はいいですか？
さあ、(f･5)キーを押してみてください。
すると、ドライブ2へディスクをセットしたかどうかの確認があり、続いて見出しの表示、コード入力となります。
コード「104」を入力しリターンキーを押すと、「GGGGG」さんでよいかどうか聞いてきますから、「Y」を入力してください。
続いてデータ修正の画面になりますから、上に示したデータに変更してください。
この過程を示す画面を次ページにかかげます。

```
ｼｭｳｾｲﾅｹﾚﾊﾞ  RET

ﾈﾝﾚｲ･････････  32
    ==> 21
ｾｯｸｽ<m/f>････m
    ==> f
ｼｮｸｷﾞｮｳ･･････2
    ==> 3
ﾅﾏｴ･･････････GGGGG
    ==>? ABCDE
ﾕｳﾋﾞﾝﾊﾞﾝｺﾞｳ･･777
    ==>? 123
ｼﾞｭｳｼｮ･･･････gggggggggg
    ==>? abcdefghij
ﾊﾞﾝﾁ･････････1-4
    ==>? 1-2-3

       Sure < y/n >?
```

修正データを入れると上の画面のように確認メッセージが現れますから、よければ「Y」を入力してください。そしてここでは、1名の修正で処理を終りにします。

次にデータが正しく修正されたかどうか確認してみたいと思いますが、110ページでセーブしたテスト用のプログラムをロードしてください。

```
load "DMTEST" ⏎
```

ロードできたらこのプログラムをRUNさせます。「Ok」が表示されたらダイレクトモードでGETして調べてみてください。次ページに入力例を示しますが、この画面では修正データだけを確認しました。最後にCLOSEするのをお忘れなく。

あとはリナンバリングしてプログラムをセーブしておきましょう。125ページから、リナンバリング後のプログラムリストをかかげます。

```
get #1,2
Ok
? sex$(7);cvi(old$(7));prof$(7)
f 21 3
Ok
get #2,14
Ok
? nm$(1):? pn$(1):? ad$(1):?ba$(1)
ABCDE
123
abcdefghij
1-2-3
Ok
get #3,1
Ok
?cvi(q1$);cvi(q2$);cvi(q3$)
 7  3  4
Ok
close
Ok
```

性別、年齢、職種

氏名、郵便番号、住所、番地

女性が１人ふえた

男性が１人減った

```
renum 100,100
Ok
renum 2000,1320
Ok
renum 3000,2070
Ok
renum 4000,3070
Ok
renum 5000,4050
Ok
renum 6000,5110
Ok
renum 7000,6080
Ok
save "DM02/2.bas"
Ok
```

プログラムリスト

```
10 ' file name ==> DM02/2.bas
20 ' 1984.1.20
30 ' random data correct
40 ' save "DM02/2.bas"
50 '
100 CONSOLE 0,25,0,0 : WIDTH 80,25
110 DEFINT A-Z
120 DEF FNSEC1(X)=(X-1)¥50+2
130 DEF FNREC1(X)=(X-1) MOD 50+1
140 DEF FNSEC2(X)=(X-1)¥3+12
150 DEF FNREC2(X)=(X-1) MOD 3+1
160 DIM NM$(3),PN$(3),AD$(3),BA$(3)
170 DIM SEX$(50),OLD$(50),PROF$(50),SPARE$(50)
180 '
190 CLS
200 LOCATE 25,4 : PRINT "ﾄﾞﾗｲﾌﾞ2ﾍ ﾃﾞｨｽｸｦﾄﾞｳｿﾞ"
210 LOCATE 25,6 : PRINT "     Sure <y/n> ?"
220 Y$=INKEY$ : IF Y$="" THEN GOTO 220
230 IF Y$="Y" OR Y$="y" THEN GOTO 250
                        ELSE GOTO 200
240 '
250 CLS
260 GOSUB *READ.INDX : FLAG=1
270 DIM CO(COUNT) : COUNT=0
280 GOSUB *READ.INDX
290 CLOSE #3
300 GOSUB *OPNF03
310 GET #1,1
320 M.COUNT=CVI(Q2$)
330 F.COUNT=CVI(Q3$)
340 CLOSE #1
350 GOSUB *OPNF01
360 GOSUB *OPNF02
370 '
380 LOCATE 25,3 : PRINT "ﾃﾞｰﾀ ﾉ ｼｭｳｾｲ ｦｼﾏｽ"
390 FOR I=1 TO 3000 : NEXT I
400 '
410 *LOOP
420 CLS
```

```
430 LOCATE 20,4 : INPUT "ｺｰﾄﾞ<END=999>";CD
440 IF CD=999 THEN GOTO *FCLOSE
450 GOSUB *DAT.SEARCH
460 IF FOUND=-1 THEN GOTO *LOOP
470 SEC1=FNSEC1(FOUND)
480 REC1=FNREC1(FOUND)
490 SEC2=FNSEC2(FOUND)
500 REC2=FNREC2(FOUND)
510 GET #1,SEC1 : GET #2,SEC2
520 '
530 CLS
540 LOCATE 20,4 : PRINT NM$(REC2)
550 LOCATE 25,5 : PRINT "ｻﾝ ﾉ ｼｭｳｾｲ <y/n>?"
560 Y$=INKEY$ : IF Y$="" THEN GOTO 560
570 IF Y$="Y" OR Y$="y" THEN GOTO *CORRECT
580 IF Y$="N" OR Y$="n" THEN GOTO *LOOP
590 BEEP : BEEP :GOTO 560
600 '
610 *CORRECT
620 CLS
630 LOCATE 20,2 : PRINT "ｼｭｳｾｲﾅｹﾚﾊﾞ RET"
640 LOCATE 20,4
650 PRINT "ﾈﾝﾚｲ･････････";CVI(OLD$(REC1))
660 LOCATE 20,5 : INPUT "    ==>";A0
670 B0=CVI(OLD$(REC1))
680 IF A0=0 THEN A0=CVI(OLD$(REC1))
690 LOCATE 20,6
700 PRINT "ｾｯｸｽ<m/f>････";SEX$(REC1)
710 LOCATE 20,7 : INPUT "    ==>";A1$
720 B1$=SEX$(REC1)
730 IF A1$="" THEN A1$=SEX$(REC1) ELSE IF
    A1$="m" OR A1$="M" THEN M.FLAG=1
    ELSE F.FLAG=1
740 GOSUB *PROFESSION
750 LOCATE 20,10
760 PRINT "ﾅﾏｴ･･････････";NM$(REC2)
770 LOCATE 20,11 : INPUT "    ==>";A2$
780 IF A2$="" THEN A2$=NM$(REC2)
790 LOCATE 20,12
800 PRINT "ﾕｳﾋﾞﾝﾊﾞﾝｺﾞｳ･･";PN$(REC2)
810 LOCATE 20,13 : INPUT "    ==>";A3$
820 IF A3$="" THEN A3$=PN$(REC2)
```

```
830 LOCATE 20,14
840 PRINT "ｼﾞｭｳｼｮ・・・・・・・";AD$(REC2)
850 LOCATE 20,15 : INPUT "    ==>";A4$
860 IF A4$="" THEN A4$=AD$(REC2)
870 LOCATE 20,16
880 PRINT "ﾊﾞﾝﾁ・・・・・・・・・";BA$(REC2)
890 LOCATE 20,17 : INPUT "    ==>";A5$
900 IF A5$="" THEN A5$=BA$(REC2)
910 LOCATE 25,19 : PRINT "Sure < y/n >?"
920 Y$=INKEY$ : IF Y$="" THEN GOTO 920
930 IF Y$="Y" OR Y$="y" THEN *DAT.WRITE1
940 IF Y$="N" OR Y$="n" THEN *LOOP
950 BEEP : BEEP :GOTO 920
960 '
970 *DAT.WRITE1
980 LSET SEX$(REC1) =A1$
990 LSET OLD$(REC1) =MKI$(A0)
1000 LSET PROF$(REC1)=PF$
1010 LSET NM$(REC2)  =A2$
1020 LSET PN$(REC2)  =A3$
1030 LSET AD$(REC2)  =A4$
1040 LSET BA$(REC2)  =A5$
1050 PUT #1,SEC1 : PUT #2,SEC2
1060 IF M.FLAG=1 THEN M.COUNT=M.COUNT+1
 : F.COUNT=F.COUNT-1 : M.FLAG=0 : GOTO 1080
1070 IF F.FLAG=1 THEN F.COUNT=F.COUNT+1
 : M.COUNT=M.COUNT-1 : F.FLAG=0
1080 A0=0 : PF$="" : A1$=""
1090 A2$="" : A3$="" : A4$="" : A5$=""
1100 GOTO *LOOP
1110 '
1120 *FCLOSE
1130 CLOSE #1,#2,#3
1140 '
1150 *DAT.WRITE2
1160 GOSUB *OPNF03
1170 GET #1,1
1180 LSET Q2$=MKI$(M.COUNT)
1190 LSET Q3$=MKI$(F.COUNT)
1200 PUT #1,1 : CLOSE #1
1210 '
1220 CLS
```

```
1230 LOCATE 25,4 : PRINT "ﾎｶ ﾉ ｼｺﾞﾄ ｦ ｼﾏｽｶ ?"
1240 LOCATE 30,5 : PRINT "< y/n >"
1250 Y$=INKEY$ : IF Y$="" THEN GOTO 1250
1260 IF Y$="Y" OR Y$="y" THEN RUN "DMmenu.bas"
1270 IF Y$="N" OR Y$="n" THEN GOTO *FINISH
1280 BEEP : BEEP :GOTO 1230
1290 '
1300 *FINISH
1310 END
2000 '
2010 *OPNF01
2020 OPEN "2:MAIL02.dat" AS #1
2030 FOR I=1 TO 50
2040 FIELD #1,(I-1)*5 AS DUMMY1$,1 AS SEX$(I)
     ,2 AS OLD$(I),1 AS PROF$(I),1 AS SPARE$(I)
2050 NEXT I
2060 RETURN
3000 '
3010 *OPNF02
3020 OPEN "2:MAIL02.dat" AS #2
3030 FOR I=1 TO 3
3040   FIELD #2,85*(I-1) AS DUMMY2$,20 AS
NM$(I),6 AS PN$(I),39 AS AD$(I),20 AS BA$(I)
3050 NEXT I
3060 RETURN
4000 '
4010 *OPNF03
4020 OPEN "2:MAIL02.dat" AS #1
4030 FIELD #1,2 AS Q1$,2 AS Q2$,2 AS Q3$,
              2 AS Q4$,2 AS Q5$,2 AS Q6$
4040 RETURN
5000 '
5010 *READ.INDX
5020 OPEN "2:INDX02.dat" FOR INPUT AS #3
5030 WHILE NOT(EOF(3))
5040   COUNT=COUNT+1
5050   IF FLAG<>1 THEN GOTO 5070
5060   INPUT #3,CO(COUNT) : GOTO 5080
5070   INPUT #3,DUMMY
5080 WEND
5090 CLOSE #3
5100 RETURN
```

```
6000 '
6010 *DAT.SEARCH
6020 FOUND=SEARCH(CO,CD)
6030 IF FOUND<>-1 THEN RETURN
6040 LOCATE 20,6 : BEEP : BEEP
6050 PRINT "ﾄｳﾛｸ ｼﾃ ｱﾘﾏｾﾝ!!"
6060 FOR I=1 TO 2000 : NEXT I
6070 RETURN
7000 '
7010 *PROFESSION
7020 CLS
7030 LOCATE 25,3
7040 PRINT "ｼｮｸｷﾞｮｳ······";PROF$(REC1)
7050 LOCATE 25,6 : PRINT "ｶｲｼｬ ﾔｸｲﾝ·····1"
7060 LOCATE 25,7 : PRINT "ｻﾗﾘｰﾏﾝ········2"
7070 LOCATE 25,8 : PRINT "ｶﾞｸｾｲ·········3"
7080 LOCATE 25,9 : PRINT "ｼｭﾌ···········4"
7090 LOCATE 25,10: PRINT "ｿﾉﾀ···········5"
7100 LOCATE 25,4 : INPUT "    ==>";PF$
7110 B2$=PROF$(REC1)
7120 IF PF$="" THEN PF$=PROF$(REC1)
7130 CLS
7140 LOCATE 20,2 : PRINT "ｼｭｳｾｲﾅｹﾚﾊﾞ RET"
7150 LOCATE 20,4 : PRINT "ﾈﾝﾚｲ·········";B0
7160 LOCATE 20,5 : PRINT "    ==>"A0
7170 LOCATE 20,6 : PRINT "ｾｯｸｽ<m/f>····";B1$
7180 LOCATE 20,7 : PRINT "    ==> ";A1$
7190 LOCATE 20,8 : PRINT "ｼｮｸｷﾞｮｳ······";B2$
7200 LOCATE 20,9 : PRINT "    ==> ";PF$
7210 RETURN
```

〈注〉290行のＣＬＯＳＥ文は、すでにサブルーチンでＣＬＯＳＥが実行されていますので不要でした。もっともＣＬＯＳＥ文はつけておいても別にエラーにはなりませんが……。

2·5　データ更改プログラム

データの新規書き込み、データ修正のプログラムができました。どうですか？　プログラミングというのは、やってみるとけっこうおもしろいと思いませんか？

エラーメッセージがでたり、正しいデータが得られなかったりしたとき、その原因を探すのも案外楽しいものです。

そして、そんなことをしているうちに、思いがけないテクニックに気がつき、だんだんと体得できていくものなのです。

ディスクの扱いにしても、前著では、ごく当たりまえの方法でしかデータの出し入れをしませんでしたが、この本を読んでみて、型にはめずに、自由に扱えることが徐々におわかりいただけたと思います。そして、この、制限にとらわれないメリットがあるからこそ、ランダムファイルなのです。

この本で、いろいろな方法をマスターし、あなたの用途・目的にあったプログラム作りをするよう努力してください。

前おきはこのくらいにして、本節では、すでに書き込んであるデータを新しくする方法を学びます。

前節の修正プログラムもデータを改めることに変わりありませんが、ここでは、数値データを書き換えてみます。

私たちがこれまでに書き込んだランダムファイルには、何人かの人の年齢が保存されています。人は時間がたてば年をとるのは自然の摂理、このランダムファイルに登録してある人も例外ではありません。年はとりたくないけど、これぱかりはどうにもなりません。

と、すれば、私たちのデータも、１年ごとに更改していかなければ、データの意味が失われてしまいますね。

そこで、毎年４月１日に１歳ずつくり上げることにします。普通は誕生日がきたら、ということになりますが、ここでは、どの人も４月１日を誕生日にしてしまいましょう。

高校生、大学生、30歳未満の社会人など、年齢制限によりダイレクトメールを発送する場合、いま述べたくらいの基準でも、それほど狂いはないと思います。

プログラムはきわめて単純なものですが、ランダムファイルでの数値データの扱いの基本が含まれていますので、しっかりマスターするよう心がけてください。

すでに書き込まれているデータはストリングですので、これを数値に変更し、それに１を加えた結果を再びもとの位置に書き込めばよいのです。プログラムの流れは、一口にいうとこんなふうに要約できるかと思います。

初期設定

この部分は、前節までとほとんど変わりません。プログラムを次に示します。

```
10  '  file name ==> DM02/3.bas
20  '  1984.1.20
30  '  increment age
40  '  save "DM02/3.bas
50  '
100 CONSOLE 0,25,0,0 : WIDTH 80,25
110 DEFINT A-Z
120 DIM SEX$(50),OLD$(50),PROF$(50),SPARE$(50)
130 DIM AGE(50)
140 '
150 CLS
160 LOCATE 25,4 : PRINT "ﾄﾞﾗｲﾌﾞ2ﾍ ﾃﾞｨｽｸｦﾄﾞｳｿﾞ"
170 LOCATE 25,6 : PRINT "    Sure <y/n> ?"
180 Y$=INKEY$ : IF Y$="" THEN GOTO 180
190 IF Y$="Y" OR Y$="y" THEN 210 ELSE 160
```

前のプログラム名とここだけちがう（行10・40の「3」）

更新した年齢を一時保存する配列（行130の AGE(50)）

くれぐれもファイル名は間違わないようにしてください。私はひどい目にあった経験があります。

行番号130の配列は、コメントにあるように、前のデータに１だけ加算した値を一時的に保存しておくためのものです。

前回更新日の確認

１年に１度更新するのでしたら間違わないと思いますが、このようなこともできるということを知っていただくために入れておきました。

プログラムは次ページに示します。

まずランダムファイルをOPENし、第１セクタをGETし、そこに書き込んであった年月日を画面に表示し、確認させます。

```
200 '
210 CLS
220 GOSUB *OPNF02
230 GET #1,1
240 TOTAL=CVI(Q1$)
250 YY=CVI(Q4$) : MM=CVI(Q5$) : DD=CVI(Q6$)
260 CLOSE #1
270 LOCATE 25,4
280 PRINT USING "####年##月##日 ニ";YY,MM,DD
290 LOCATE 30,5 : PRINT "コウシンズミ デス"
300 LOCATE 25,7 : PRINT "コウシン シマスカ <y/n> ?"
310 Y$=INKEY$ : IF Y$="" THEN GOTO 310
320 IF Y$="Y" OR Y$="y" THEN *INCREMENT
330 IF Y$="N" OR Y$="n" THEN *FINISH
340 BEEP : BEEP :GOTO 310
```

登録データ数の数値変換

年月日の数値変換

第2章で作っている一連のプログラムでは、ランダムファイルの第1セクタの頭から登録データ数、男性の登録数、女性の登録数、年、月、日と順に、各2バイトずつ使っていました。

そして、年月日は書き込みをするときいつも入力していましたが、ベースとなる年月日は一度だけでよいわけですから、ある年度の最初には期首年月日とデータ入力日を、その年度内でのデータ追加のときはデータ入力日だけを、ということにして2つの日付を第1セクタに書き込むようにすると、もっと意味がでてくるかもしれません。

第1セクタはまだまだたくさんアキがあるのですから、有効な使い方をみなさん考えてみてください。

240行で登録データ数を数値に変換してメモリ上に呼び出していますが、これはあとでＧＥＴするセクタ計算に使います。

220行にGOSUB文がありますので、ファイルOPENとFIELD設定のサブルーチンをキーインしておきましょう。

いつもとまったく同じです。

```
2000 '
2010 *OPNF02
2020 OPEN "2:MAIL02.dat" AS #1
2030 FIELD #1,2 AS Q1$,2 AS Q2$,2 AS Q3$,
              2 AS Q4$,2 AS Q5$,2 AS Q6$
2040 RETURN
```

もう1つ、性別、年齢などのフィールド設定をする必要がありますので、次のサブルーチンもキーインしておきます。

```
1000 '
1010 *OPNF01
1020 OPEN "2:MAIL02.dat" AS #1
1030 FOR I=1 TO 50
1040 FIELD #1,(I-1)*5 AS DUMMY1$,1 AS SEX$(I)
     ,2 AS OLD$(I),1 AS PROF$(I),1 AS SPARE$(I)
1050 NEXT I
1060 RETURN
```

年齢加算と書き込み部分

すでに書き込まれている年齢を数値変換してメモリ上に取り出し、1だけ加算して再びPUTするルーチンです。プログラムを次に示します。

```
350 '
360 *INCREMENT
370 YY=YY+1                          ← 期首年も更新
380 GOSUB *OPNF01
390 S.LENGTH=(TOTAL-1)¥50+2          ← データが書き込まれている最終セクタを計算
400 FOR I=2 TO S.LENGTH
410   GET #1,I
420   FOR J=1 TO 50
430     PNT=PNT+1                    ← レコード番号を1ずらす
440     AGE(PNT)=CVI(OLD$(PNT))+1    ← プラス1する
450   NEXT J
460   FOR J=1 TO 50
470     LSET OLD$(J)=MKI$(AGE(J))    ← 更新データをLSET
480   NEXT J
490   PUT #1,I                       ← データの書き込み
500   PNT=0
510 NEXT I
520 CLOSE #1
```

370行では、期首年もいっしょにプラス1してしまいました。

390行は、第2セクタから第11セクタまでのどのセクタまでデータが書き込まれているか計算するステートメントです。これをやっておかないと、第11セクタまでGETすることになり、だいぶムダな仕事

をすることになってしまいます。

原理は、初項を２として、50レコード経過するたびにセクタを１進めるにはどう計算すればよいか、ということです。

この種の計算は、前にセクタ番号、そのセクタ内でのレコード番号を計算するときにやりましたのでわかりますね。

この部分ではFOR～NEXTループを２回使っています。

外側のループはセクタを１つ進めるために、内側のループはレコード番号を１きざみに50だけ進めるために設けました。

外側のループを実行するたびに、410行でそのセクタをＧＥＴしています。その後、内側のループに入りますが、まずポインタの役割を果たす変数としてＰＮＴを作り、430行で１ずつ加算するようにします。つまり、これが、あるデータのそのセクタ内でのレコード番号になるわけですね。

440行では、年齢をプラス１しています。

460～480行のループでは、いま更新した年齢データをバッファ上でＬＳＥＴしています。そして490行のＰＵＴ文でディスクに書き込みます。

ＰＮＴをゼロクリアするのを忘れないでください。これをやっておかないと、２回目のＩのループに入ったとき、430行でこの値が51になりますから、次の440行で

```
Subscript out of range in 440
```

のエラーになると思います。

大きなループを抜けたら、ファイルをCLOSEしておきましょう。

以上で、年齢更新、書き込みのルーチンは終りです。もう一度この部分をながめてみてください。

間違いないように見えるでしょう。しかし、１つ欠陥があるのです。それは、配列ＡＧＥを、セクタを１つ進めるごとにクリアしてないことです。

私たちの場合には、第２セクタしかまだ書き込んでありませんから、このプログラムで正常に走ると思いますが、第３セクタにも51人目からのデータが書き込んであるとすると、第２セクタでの１人目から51人目までのデータが配列の中にそのまま残っていますから、第３セクタの１人目の場合、第２セクタの１人目の年齢にその人の年齢を加え、さらに１プラスすることになり、場合によっては100歳以上の人が続出することになります。

次のステートメントを入れておきましょう。

```
495   FOR J=1 TO 50 : AGE(J)=0 : NEXT J
```

年齢の更改が終りましたら、今度は第1セクタへの年月日の書き込みです。ランダムファイルは閉じられていますから、もう一度オープンしなければなりませんね。

次のステートメントをキーインしてください。

```
530 '
540 GOSUB *OPNF02
550 GET #1,1
560   LSET Q4$=MKI$(YY)
570   LSET Q5$=MKI$(MM)
580   LSET Q6$=MKI$(DD)
590 PUT #1,1
600 CLOSE #1
```

最後にエンディング部分。これはもうおなじみですね。

```
610 '
620 CLS
630 LOCATE 25,4 : PRINT "ｼｮﾘ ｦ ｵﾜﾘﾏｽ"
640 LOCATE 25,5 : PRINT "ﾎｶ ﾉ ｼｺﾞﾄ ｦ ｼﾏｽｶ ?"
650 LOCATE 30,7 : PRINT "< y/n >"
660 Y$=INKEY$ : IF Y$="" THEN GOTO 660
670 IF Y$="Y" OR Y$="y" THEN RUN "DMmenu.bas"
680 IF Y$="N" OR Y$="n" THEN GOTO *FINISH
690 BEEP : BEEP :GOTO 640
700 '
710 *FINISH
720 END
```

そして、このプログラムのセーブです。

```
save "DM02/3.bas"
```

プログラムの実行

セーブできましたら、このプログラムをさっそくＲＵＮさせてみましょう。

（f・5）キーを押しますと、いつものようにデータディスクセット

の確認があり、続いて年齢の更新をするかどうか聞いてきます。
「Y」を入力すると、たいした時間もかからずに処理が終了します。

それでは、更新できたかどうか確かめてみたいと思いますが、テストプログラムをロードしてください。

```
load "DMTEST"
```

そしてRUNさせ、次の画面のようにダイレクトモードで入力し確認してください。

```
run
Ok
get #1,2
Ok
for i=1 to 7:? cvi(old$(i)):next
 43
 11
 51
 28
 31
 20
 22
Ok
get #3,1
Ok
? cvi(q4$):? cvi(q5$): ? cvi(q6$)
 1985
 4
 1
Ok
close
Ok
█
```

テストプログラムの実行（run）
更新データ（43〜22）
●109ページのデータと比較してみてください
年度も更新されている（1985）
これを忘れないように（close）

成功したようです。あなたはどうでしたか？
もしうまくいきませんでしたら、上の画面のＧＥＴするバッファ番号、セクタが違っていないか、あるいはプログラム中のファイル番号が違っていないかなどをチェックしてください。
次のページに、プログラムリストをのせておきます。

プログラムリスト

```
10 '  file name ==> DM02/3.bas
20 '  1984.1.20
30 '  increment age
40 '  save "DM02/3.bas"
50 '
100 CONSOLE 0,25,0,0 : WIDTH 80,25
110 DEFINT A-Z
120 DIM SEX$(50),OLD$(50),PROF$(50),SPARE$(50)
130 DIM AGE(50)
140 '
150 CLS
160 LOCATE 25,4 : PRINT "ﾄﾞﾗｲﾌﾞ2ﾍ ﾃﾞｨｽｸｦﾄﾞｳｿﾞ"
170 LOCATE 25,6 : PRINT "     Sure <y/n> ?"
180 Y$=INKEY$ : IF Y$="" THEN GOTO 180
190 IF Y$="Y" OR Y$="y" THEN 210 ELSE 160
200 '
210 CLS
220 GOSUB *OPNF02
230 GET #1,1
240 TOTAL=CVI(Q1$)
250 YY=CVI(Q4$) : MM=CVI(Q5$) : DD=CVI(Q6$)
260 CLOSE #1
270 LOCATE 25,4
280 PRINT USING "####年##月##日 ﾆ";YY,MM,DD
290 LOCATE 30,5 : PRINT "ｺｳｼﾝｽﾞﾐ ﾃﾞｽ"
300 LOCATE 25,7 : PRINT "ｺｳｼﾝ ｼﾏｽｶ <y/n> ?"
310 Y$=INKEY$ : IF Y$="" THEN GOTO 310
320 IF Y$="Y" OR Y$="y" THEN *INCREMENT
330 IF Y$="N" OR Y$="n" THEN *FINISH
340 BEEP : BEEP :GOTO 310
350 '
360 *INCREMENT
370 YY=YY+1
380 GOSUB *OPNF01
390 S.LENGTH=(TOTAL-1)¥50+2
400 FOR I=2 TO S.LENGTH
410   GET #1,I
420   FOR J=1 TO 50
430     PNT=PNT+1
440     AGE(PNT)=CVI(OLD$(PNT))+1
```

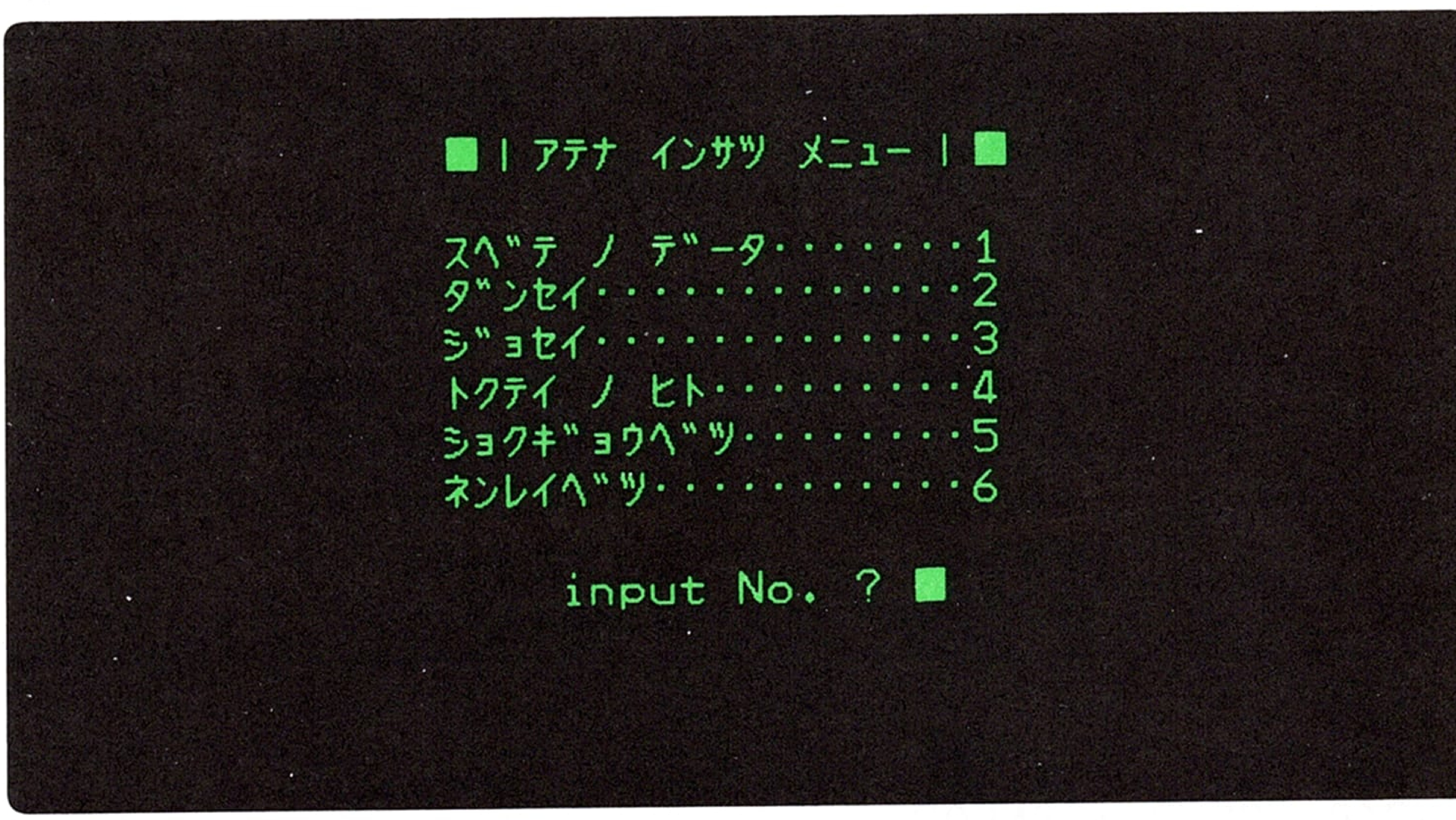
```
450    NEXT J
460    FOR J=1 TO 50
470      LSET OLD$(J)=MKI$(AGE(J))
480    NEXT J
490    PUT #1,I
495    FOR J=1 TO 50 : AGE(J)=0 : NEXT J
500    PNT=0
510 NEXT I
520 CLOSE #1
530 '
540 GOSUB *OPNF02
550 GET #1,1
560   LSET Q4$=MKI$(YY)
570   LSET Q5$=MKI$(MM)
580   LSET Q6$=MKI$(DD)
590 PUT #1,1
600 CLOSE #1
610 '
620 CLS
630 LOCATE 25,4 : PRINT "ｼｮﾘ ｦ ｵﾜﾘﾏｽ"
640 LOCATE 25,5 : PRINT "ﾎｶ ﾉ ｼｺﾞﾄ ｦ ｼﾏｽｶ ?"
650 LOCATE 30,7 : PRINT "< y/n >"
660 Y$=INKEY$ : IF Y$="" THEN GOTO 660
670 IF Y$="Y" OR Y$="y" THEN RUN "DMmenu.bas"
680 IF Y$="N" OR Y$="n" THEN GOTO *FINISH
690 BEEP : BEEP :GOTO 640
700 '
710 *FINISH
720 END
1000 '
1010 *OPNF01
1020 OPEN "2:MAIL02.dat" AS #1
1030 FOR I=1 TO 50
1040 FIELD #1,(I-1)*5 AS DUMMY1$,1 AS SEX$(I)
     ,2 AS OLD$(I),1 AS PROF$(I),1 AS SPARE$(I)
1050 NEXT I
1060 RETURN
2000 '
2010 *OPNF02
2020 OPEN "2:MAIL02.dat" AS #1
2030 FIELD #1,2 AS Q1$,2 AS Q2$,2 AS Q3$,
              2 AS Q4$,2 AS Q5$,2 AS Q6$
2040 RETURN
```

2・6　6つの印刷メニュー

私たちはこの章で、第1章の場合よりさらに項目をふやしてデータファイルを作ってきました。

そこで、まとめとして、いろいろな条件別に宛て名を出力するプログラムを作ってみましょう。出力方法は、プリンタをお持ちでない方もいると思いますので、第1章と同じように、ディスプレイにも出力できるようにしておきましょう。

メニュー画面は、職業別、年齢別を加えて、次のようにしてみました。



初期設定をし、上のようなメニュー画面を表示して希望の番号を入力し、その番号によって各サブルーチンへ分岐して印刷すべきデータをチェックし、その登録順位を配列にしまい、メインルーチンに戻って印刷するというのが、本章でのプログラムの大きな流れです。

第1章では、特定の人への宛て名印刷は別のプログラムを走らせるようにしましたが、ここでは、6つのメニュー処理を1つのプログラムにおさめてしまうようにくふうしました。

したがって、プログラムは少し長くなりますが、配列でそれほどメモリを使わないので、容量面からいっても十分余裕があります。

それではプログラミングに入りますが、データ修正プログラムの一部が使えますので、まずこれをロードして不要行を削除することにしましょう。

必要な部分は、初期設定および、職種選択部分を除くサブルーチンですので、不要行をデリートするまでの操作は、次の画面に示すようになります。

```
load "DM02/2.bas"
Ok
delete 240-1200
Ok
delete 7000-7210
Ok
■
```

ここまでの操作が終りましたら次へ進みますが、一度リストコマンドで残っているプログラムリストを画面に表示し、これを頭に描いてから読み進めてください。

配列宣言とファイルオープン

このプログラムでは、出力すべきデータの登録順位を一時配列に格納し、あとでまとめて出力しますので、配列を宣言する必要がありますが、この大きさは、ランダムファイルの第1セクタにデータの総数が書き込まれていますので、これを使うことにします。

この部分のプログラムを次に示します。

```
240 '
250 CLS
260 GOSUB *OPNF03          ←ランダムファイルオープン
270 GET #1,1
280 TOTAL=CVI(Q1$)         ←データ総数をメモリに取り出す
290 CLOSE #1
300 DIM B(TOTAL),CO(TOTAL) ←データ総数を使って配列宣言
305 GOSUB *READ.INDX       ←配列内にコードを読み込む
310 GOSUB *OPNF01 : GOSUB *OPNF02
320 '
330 LOCATE 25,3 : PRINT "ｱﾃﾅ ﾉ ｲﾝｻﾂｦｼﾏｽ"
340 FOR I=1 TO 2000 : NEXT I
```

データ修正プログラムでは、一度シーケンシャルファイルをカラ読みして配列を宣言していましたが、上のようにすると、処理時間もだいぶ節約できます。

310行では、第2セクタ以降のデータをバッファ上に呼び出せるよ

うにランダムファイルをOPENし、フィールドを設定しています。

配列宣言で、データの読み飛ばしが不要になりましたので、5050行および5070行は削除し、5060行はGOTO文をとって次のように修正しておきましょう。

```
5050 ↵
5070 ↵
```

```
5060    INPUT #3,CO(COUNT)
```

メニュー画面

この部分のプログラムを次に示します。

```
350 '
360 *MENU
370 CLS
380 LOCATE 25,3 : PRINT "■|ｱﾃﾅ ｲﾝｻﾂ ﾒﾆｭｰ|■"
390 LOCATE 25,5 : PRINT "ｽﾍﾞﾃ ﾉ ﾃﾞｰﾀ･･･････1"
400 LOCATE 25,6 : PRINT "ﾀﾞﾝｾｲ････････････2"
410 LOCATE 25,7 : PRINT "ｼﾞｮｾｲ････････････3"
420 LOCATE 25,8 : PRINT "ﾄｸﾃｲ ﾉ ﾋﾄ･････････4"
430 LOCATE 25,9 : PRINT "ｼｮｸｷﾞｮｳﾍﾞﾂ･･･････5"
440 LOCATE 25,10: PRINT "ﾈﾝﾚｲﾍﾞﾂ･･･････････6"
450 LOCATE 25,12: INPUT "    input No.";NO
460 ON NO GOSUB *P1,*P2,*P3,*P4,*P5,*P6
470 '
480 CLS
490 LOCATE 25,5
500 PRINT "ﾌﾟﾘﾝﾀ ｦ ﾓｯﾃ ｲﾏｽｶ <y/n> ?"
510 Y$=INKEY$ : IF Y$="" THEN GOTO 510
520 IF Y$="Y" OR Y$="y" THEN P.FLAG=1
    :GOTO *DAT.PRT
530 IF Y$="N" OR Y$="n" THEN GOTO *DAT.PRT
540 BEEP : BEEP :GOTO 510
```

460行で、キーインされたメニュー番号に応じて、各サブルーチンへ分岐させています。**サブルーチンの役割は、出力すべきデータの、ファイルに書き込まれている順位（先頭から何番目のデータかという通し番号〈注〉）を配列に格納することと、出力するデータ数を調べることです。**

〈注〉ここでいう通し番号とは、109ページのデータで、たとえば「ＥＥＥＥＥ」さんなら５、「ＧＧＧＧＧ」さんなら７ということです。

サブルーチンから戻ったらプリンタの有無をチェックし、出力ルーチンへジャンプさせます。

プリントアウトルーチン

サブルーチンを作る前に、このほうがわかりやすいので宛て名の印刷・表示のルーチンを完成させておきましょう。

```
550 '
560 *DAT.PRT
570 IF P.FLAG<>1 THEN GOTO 630
580 CLS : LOCATE 25,5
590 PRINT "ﾌﾟﾘﾝﾀ ｦ ｾｯﾄ ｼﾃ ｸﾀﾞｻｲ <y/n> ?"
600 Y$=INKEY$ : IF Y$="" THEN GOTO 600
610 IF Y$="Y" OR Y$="y" THEN GOTO 660
620 BEEP : BEEP :GOTO 600
630 CLS : LOCATE 25,5
640 PRINT "ｱﾃﾅ ｦ ﾋｮｳｼﾞ ｼﾏｽ"
650 FOR I=1 TO 2000 : NEXT I
660 FOR I=1 TO NUMBER
670   IF B(I)=0 THEN P=I : GOTO 690
680   P=B(I)
690   SEC2=FNSEC2(P)
700   REC2=FNREC2(P)
710   GET #2,SEC2
720   IF P.FLAG=1 THEN GOTO 820
730   CLS
740   LOCATE 25,5 : PRINT NM$(REC2)
750   LOCATE 25,6 : PRINT PN$(REC2)
760   LOCATE 25,7 : PRINT AD$(REC2)
770   LOCATE 25,8 : PRINT BA$(REC2)
780   LOCATE 20,10
790   PRINT "Hit RET key for next person !"
800   Y$=INKEY$ : IF Y$="" THEN GOTO 800
810   GOTO 880
820   LPRINT PN$(REC2)
830   LPRINT AD$(REC2)
840   LPRINT BA$(REC2)
850   LPRINT TAB(5);NM$(REC2)
860   LPRINT TAB(15);"ｻﾏ"
870   FOR J=1 TO 5 : LPRINT : NEXT J
880 NEXT I
```

550行から650行まではおなじみのステートメントですから問題ないでしょう。

660 行からが実際の印刷部分ですが、出力するデータを各サブルーチンでカウントし、変数NUMBERに代入しておき、FOR～NEXTループでこの数だけくりかえし処理を行います。

670 行は、全員の宛て名を印刷するときのステートメントです。つまり、配列Bには印刷データの通し番号が格納してあるはずですが、全員の印刷のときには先頭データから1ステップずつずらしてくりかえせばよいので、ループ回数（全データ数）さえわかれば、配列を使う必要がありません。

ということは、この場合には配列Bの中味はすべて0になっています。ですから、この条件を満たす場合には、ループ変数の中味をPに代入し、セクタ計算のステートメントへジャンプさせればよいのです。

全員のデータを印刷するだけでしたら、670行も680行も必要ないのですが、できたら6つのメニュー選択に対して同じ印刷ルーチンを使いたいので、このようにしました。

つまりPは、このルーチンを共用するためにわざわざ用意した変数ということになります。

680 行では、配列にしまわれている印刷データの通し番号をPに代入し、690行、700行で、DEF FN文で定義した関数を使って、セクタ番号、そのセクタ内でのレコード番号を計算しています。

その他のステートメントについては、第1章でもやりましたので、ご理解いただけると思います。

次に、各サブルーチンを作っていきますが、いま述べたように、この印刷ルーチンを共用することが前提ですので、このことを頭において進めていきましょう。

全データ印刷のサブルーチン

この部分は簡単です。登録データ数を 280 行でランダムファイルから取り出してありましたから、これをNUMBERに代入して戻せばよいのです。

```
7000 '
7010 *P1
7020 NUMBER=TOTAL
7030 RETURN
```

男女別のサブルーチン

まずプログラムを示します。

```
7200 '
7210 *P2
7220 OBJ1$="m" : OBJ2$="M"
7230 GOSUB *PICKUP
7240 RETURN
7400 '
7410 *P3
7420 OBJ1$="f" : OBJ2$="F"
7430 GOSUB *PICKUP
7440 RETURN
```

（7210～7240：男性の場合　7410～7440：女性の場合）

この場合にはまず、データを抽出するためのチェッカーを変数に代入します。7220行、7420行がその処理です。そして、さらにサブルーチンにジャンプさせて印刷データを選び出すわけですが、このようにしておくと、男性、女性の場合とも同じデータ抽出サブルーチンが使えます。

次に、その抽出サブルーチンを示します。

```
9000 '
9010 *PICKUP
9020 S.LENGTH=(TOTAL-1)¥50+2
9030 FOR I=2 TO S.LENGTH
9040  GET #1,I
9050  FOR J=1 TO 50
9060   PNT=PNT+1
9070   IF SEX$(PNT)=OBJ1$ OR SEX$(PNT)=OBJ2$
       THEN GOTO 9080 ELSE GOTO 9090
9080   TOROKU=TOROKU+1:B(TOROKU)=(I-2)*50+PNT
9090  NEXT J
9100  PNT=0
9110 NEXT I
9120 NUMBER=TOROKU
9130 RETURN
```

この部分は133ページのプログラムと形が似ていますのでわかると思いますが、簡単に説明しておきます。

まず9020行で、全データ数をもとにして、性別などが書き込んであ

るセクタをどこまで使用しているか、その最終セクタを算出します。

そして、第２セクタから最終セクタまでループさせ、その内側にセクタ内でレコード番号を１ステップずつシフトするループを作り、各レコードの性別用のチェッカーを調べます。

9040行では、セクタが１つ進むごとに、新しいセクタをＧＥＴしています。また、9060行では、レコード番号を１ずつカウントし、9070行で、その位置の性別用チェッカーを調べています。

9080行は、該当データがあった場合の登録処理です。まず登録用カウンタを１つ上げて、先頭データ（第２セクタの第１番目のデータ）から何番目かを計算し、その値を配列にしまっておきます。

新しいセクタのループに行く前に、9100行でやっているように、カウンタをクリアするのを忘れないでください。

そして、外側のループから抜け出たときに、登録数をNUMBERに代入してこのサブルーチンからもとの場所へ戻ります。

こうしておけば、先ほどの印刷ルーチンがそのまま使えますね。

●9030行のＦＯＲ文でのループ変数は２から始まりますので注意してください。１からループする習慣がついていますので、ついうっかりこれを１にしますと、9040行で第１セクタをＧＥＴしてしまいますから、正しいデータが得られません。

●9060行のＰＮＴはレコードのポインタのつもりでしたが、これはループ変数Ｊで代用できそうです。この場合、9060行を削除し9070行と9080行のＰＮＴをＪにすればよいでしょう。そのほかのサブルーチンについても、まったく同じ考え方ができます。

コード入力によるサブルーチン

```
7600 '
7610 *P4
7620 CLS
7630 LOCATE 25,4 : INPUT "ｺｰﾄﾞ<END=999>";CD
7640 IF CD=999 THEN GOTO 7780
7650 GOSUB *DAT.SEARCH
7660 IF FOUND=-1 THEN 7600
7670 SEC2=FNSEC2(FOUND)
7680 REC2=FNREC2(FOUND)
7690 GET #2,SEC2
7700 CLS
7710 LOCATE 25,4 : PRINT NM$(REC2)
7720 LOCATE 25,6 : PRINT "ﾄｳﾛｸ ｼﾏｽｶ <y/n> ?"
7730 Y$=INKEY$ : IF Y$="" THEN GOTO 7730
7740 IF Y$="Y" OR Y$="y" THEN TOROKU=TOROKU+1
      : B(TOROKU)=FOUND : GOTO 7770
7750 IF Y$="N" OR Y$="n" THEN 7600
7760 BEEP : BEEP : GOTO 7730
7770 GOTO 7600
7780 NUMBER=TOROKU
7790 RETURN
```

この部分は問題ないと思います。

ただ、このサブルーチンから帰る前に、登録数をNUMBERに代入するのを忘れないようにしてください。

職種別のサブルーチン

このサブルーチンも、男女別のサブルーチンと理屈はまったく同じです。

職種を選択しなければいけませんので、メニュー画面を加えておきましょう。

次のようにキーインしてください。

```
8000 '
8010 *P5
8020 CLS
8030 LOCATE 25,4 : PRINT "■ |    ﾒﾆｭｰ    | ■"
8040 LOCATE 25,6 : PRINT "ｶｲｼｬ ﾔｸｲﾝ・・・・・1"
8050 LOCATE 25,7 : PRINT "ｻﾗﾘｰﾏﾝ・・・・・・・・2"
8060 LOCATE 25,8 : PRINT "ｶﾞｸｾｲ・・・・・・・・・3"
8070 LOCATE 25,9 : PRINT "ｼｭﾌ・・・・・・・・・・・4"
8080 LOCATE 25,10: PRINT "ｿﾉﾀ・・・・・・・・・・・5"
8090 LOCATE 25,12: INPUT "  input No.";OBJ$
8100 S.LENGTH=(TOTAL-1)¥50+2
8110 FOR I=2 TO S.LENGTH
8120  GET #1,I
8130  FOR J=1 TO 50
8140   PNT=PNT+1
8150   IF PROF$(PNT)=OBJ$ THEN GOTO 8160
        ELSE GOTO 8170
8160   TOROKU=TOROKU+1:B(TOROKU)=(I-2)*50+PNT
8170  NEXT J
8180  PNT=0
8190 NEXT I
8200 NUMBER=TOROKU
8210 RETURN
```

各職種に対応する番号は、ランダムファイルへ書き込むとき、数値データとしてではなく、ストリングとして扱いましたので、8090行のINPNT文でも、キーインされた番号をストリング変数に代入しています。

年齢区分によるサブルーチン

やはりこの場合も、いまの職種別サブルーチンとメニュー画面が異なるくらいで、そのほかはほとんど同じです。

```
8400 '
8410 *P6
8420 CLS
8430 LOCATE 25,4 : PRINT "■|    ﾒﾆｭｰ   |■"
8440 LOCATE 25,6 : PRINT "30 ｻｲ ﾐﾏﾝ·····1"
8450 LOCATE 25,7 : PRINT "30 ｻｲ ｲｼﾞｮｳ···2"
8460 LOCATE 25,9 : INPUT "  input No.";AGE
8470 S.LENGTH=(TOTAL-1)¥50+2
8480 FOR I=2 TO S.LENGTH
8490  GET #1,I
8500  FOR J=1 TO 50
8510   PNT=PNT+1
8520   IF AGE=2 THEN GOTO 8540
8530   IF CVI(OLD$(PNT))>0 AND CVI(OLD$(PNT))
        <30 THEN GOTO 8550 ELSE GOTO 8560
8540   IF CVI(OLD$(PNT))>=30 THEN GOTO 8550
        ELSE GOTO 8560
8550   TOROKU=TOROKU+1:B(TOROKU)=(I-2)*50+PNT
8560  NEXT J
8570  PNT=0
8580 NEXT I
8590 NUMBER=TOROKU
8600 RETURN
```

メニューは2種類に限定しました。30歳未満と30歳以上です。

もう少し分類したいという方は、本章が終ってからチャレンジしてみてください。

8530行、8540行が年齢判定の条件文ですが、ここで注意しなくてはいけないのは、30歳未満の場合です。

30歳未満ですから、

```
CVI(OLD$(PNT))<30
```

という条件だけでよさそうに見えますが、「0より大きく」という条件もつけておきませんと、おかしなことになってしまいます。

なぜかと申しますと、ランダムファイルへの書き込み時にデータをLSETしてPUTしますと、そのセクタの残りの部分には0が書き込

まれます。たとえば、私たちの場合ですと、第2セクタには7つのデータしか書き込んでありませんから、残りの43のデータを書き込む個所には0が入ります。

そうしますと、もし「0より大きく」(0は含まれません)という条件がないと、8番目のデータから50番目のデータまですべて0ですから、どれも8530行の条件を満たすことになり、このプログラムですと8550行で、

```
Subscript out of range in 8550
```

のエラーとなると思います。

つまり、どのデータも登録されてしまうことになり、配列が不足してしま　わけです。

このプログラムでは配列を最小限にとってありましたからまだいいのですが、たとえだ配列を50くらいとってありますと、このサブルーチンではエラーにならず、印刷ルーチンへ行って、おかしな出力をすることになります。

こんなときのバグはちょっと探しにくいですよ。

これで、追加するべきサブルーチンはすべて終りました。

あとは、先頭のＲＥＭ文の修正と、印刷が終ってからファイルをＣＬＯＳＥするだけとなりました。

```
10 ' file name ==> DM02/4.bas
20 ' 1984.1.20
30 ' random data print out
40 ' save "DM02/4.bas"
50 '
```

```
890 '
900 *FCLOSE
910 CLOSE #1,#2
```

これで、本節は終りです。

このプログラムの実行についての説明は省略しますので、みなさん各自でRUNさせてみてください。

プログラム総リストを次ページからかかげますが、この節のはじめで不要行を削除したとき、残っているリストを示しませんでしたので、リナンバリングせずに掲載します。

プログラムリスト

```
10 ' file name ==> DM02/4.bas
20 ' 1984.1.20
30 ' random data print out
40 ' save "DM02/4.bas"
50 '
100 CONSOLE 0,25,0,0 : WIDTH 80,25
110 DEFINT A-Z
120 DEF FNSEC1(X)=(X-1)¥50+2
130 DEF FNREC1(X)=(X-1) MOD 50+1
140 DEF FNSEC2(X)=(X-1)¥3+12
150 DEF FNREC2(X)=(X-1) MOD 3+1
160 DIM NM$(3),PN$(3),AD$(3),BA$(3)
170 DIM SEX$(50),OLD$(50),PROF$(50),SPARE$(50)
180 '
190 CLS
200 LOCATE 25,4 : PRINT "ﾄﾞﾗｲﾌﾞ2ﾍ ﾃﾞｨｽｸｦﾄﾞｳｿﾞ"
210 LOCATE 25,6 : PRINT "     Sure <y/n> ?"
220 Y$=INKEY$ : IF Y$="" THEN GOTO 220
230 IF Y$="Y" OR Y$="y" THEN GOTO 250
                         ELSE GOTO 200
240 '
250 CLS
260 GOSUB *OPNF03
270 GET #1,1
280 TOTAL=CVI(Q1$)
290 CLOSE #1
300 DIM B(TOTAL),CO(TOTAL)
305 GOSUB *READ.INDX
310 GOSUB *OPNF01 : GOSUB *OPNF02
320 '
330 LOCATE 25,3 : PRINT "ｱﾃﾅ ﾉ ｲﾝｻﾂｦｼﾏｽ"
340 FOR I=1 TO 2000 : NEXT I
350 '
360 *MENU
370 CLS
380 LOCATE 25,3 : PRINT "■|ｱﾃﾅ ｲﾝｻﾂ ﾒﾆｭｰ|■"
390 LOCATE 25,5 : PRINT "ｽﾍﾞﾃ ﾉ ﾃﾞｰﾀ・・・・・・・・1"
400 LOCATE 25,6 : PRINT "ﾀﾞﾝｾｲ・・・・・・・・・・・・・・2"
410 LOCATE 25,7 : PRINT "ｼﾞｮｾｲ・・・・・・・・・・・・・・3"
420 LOCATE 25,8 : PRINT "ﾄｸﾃｲ ﾉ ﾋﾄ・・・・・・・・・・4"
```

```
430 LOCATE 25,9 : PRINT "ｼｮｸｷﾞｮｳﾍﾞﾂ････････5"
440 LOCATE 25,10: PRINT "ﾈﾝﾚｲﾍﾞﾂ･･･････････6"
450 LOCATE 25,12: INPUT "     input No.";NO
460 ON NO GOSUB *P1,*P2,*P3,*P4,*P5,*P6
470 '
480 CLS
490 LOCATE 25,5
500 PRINT "ﾌﾟﾘﾝﾀ ｦ ﾓｯﾃ ｲﾏｽｶ <y/n> ?"
510 Y$=INKEY$ : IF Y$="" THEN GOTO 510
520 IF Y$="Y" OR Y$="y" THEN P.FLAG=1
    :GOTO *DAT.PRT
530 IF Y$="N" OR Y$="n" THEN GOTO *DAT.PRT
540 BEEP : BEEP :GOTO 510
550 '
560 *DAT.PRT
570 IF P.FLAG<>1 THEN GOTO 630
580 CLS : LOCATE 25,5
590 PRINT "ﾌﾟﾘﾝﾀ ｦ ｾｯﾄ ｼﾃ ｸﾀﾞｻｲ <y/n> ?"
600 Y$=INKEY$ : IF Y$="" THEN GOTO 600
610 IF Y$="Y" OR Y$="y" THEN GOTO 660
620 BEEP : BEEP :GOTO 600
630 CLS : LOCATE 25,5
640 PRINT "ｱﾃﾅ ｦ ﾋｮｳｼﾞ ｼﾏｽ"
650 FOR I=1 TO 2000 : NEXT I
660 FOR I=1 TO NUMBER
670   IF B(I)=0 THEN P=I : GOTO 690
680   P=B(I)
690   SEC2=FNSEC2(P)
700   REC2=FNREC2(P)
710   GET #2,SEC2
720   IF P.FLAG=1 THEN GOTO 820
730   CLS
740   LOCATE 25,5 : PRINT NM$(REC2)
750   LOCATE 25,6 : PRINT PN$(REC2)
760   LOCATE 25,7 : PRINT AD$(REC2)
770   LOCATE 25,8 : PRINT BA$(REC2)
780   LOCATE 20,10
790   PRINT "Hit RET key for next person !"
800   Y$=INKEY$ : IF Y$="" THEN GOTO 800
810   GOTO 880
820   LPRINT PN$(REC2)
830   LPRINT AD$(REC2)
```

```
840   LPRINT BA$(REC2)
850   LPRINT TAB(5);NM$(REC2)
860   LPRINT TAB(15);"ｻﾏ"
870   FOR J=1 TO 5 : LPRINT : NEXT J
880 NEXT I
890 '
900 *FCLOSE
910 CLOSE #1,#2
1210 '
1220 CLS
1230 LOCATE 25,4 : PRINT "ﾎｶ ﾉ ｼｺﾞﾄ ｦ ｼﾏｽｶ ?"
1240 LOCATE 30,5 : PRINT "< y/n >"
1250 Y$=INKEY$ : IF Y$="" THEN GOTO 1250
1260 IF Y$="Y" OR Y$="y" THEN RUN "DMmenu.bas"
1270 IF Y$="N" OR Y$="n" THEN GOTO *FINISH
1280 BEEP : BEEP :GOTO 1230
1290 '
1300 *FINISH
1310 END
2000 '
2010 *OPNF01
2020 OPEN "2:MAIL02.dat" AS #1
2030 FOR I=1 TO 50
2040 FIELD #1,(I-1)*5 AS DUMMY1$,1 AS SEX$(I)
     ,2 AS OLD$(I),1 AS PROF$(I),1 AS SPARE$(I)
2050 NEXT I
2060 RETURN
3000 '
3010 *OPNF02
3020 OPEN "2:MAIL02.dat" AS #2
3030 FOR I=1 TO 3
3040   FIELD #2,85*(I-1) AS DUMMY2$,20 AS
NM$(I),6 AS PN$(I),39 AS AD$(I),20 AS BA$(I)
3050 NEXT I
3060 RETURN
4000 '
4010 *OPNF03
4020 OPEN "2:MAIL02.dat" AS #1
4030 FIELD #1,2 AS Q1$,2 AS Q2$,2 AS Q3$,
          2 AS Q4$,2 AS Q5$,2 AS Q6$
4040 RETURN
5000 '
```

```
5010 *READ.INDX
5020 OPEN "2:INDX02.dat" FOR INPUT AS #3
5030 WHILE NOT(EOF(3))
5040   COUNT=COUNT+1
5060   INPUT #3,CO(COUNT)
5080 WEND
5090 CLOSE #3
5100 RETURN
6000 '
6010 *DAT.SEARCH
6020 FOUND=SEARCH(CO,CD)
6030 IF FOUND<>-1 THEN RETURN
6040 LOCATE 25,6 : BEEP : BEEP
6050 PRINT "ﾄｳﾛｸ ｼﾃ ｱﾘﾏｾﾝ!!"
6060 FOR I=1 TO 2000 : NEXT I
6070 RETURN
7000 '
7010 *P1
7020 NUMBER=TOTAL
7030 RETURN
7200 '
7210 *P2
7220 OBJ1$="m" : OBJ2$="M"
7230 GOSUB *PICKUP
7240 RETURN
7400 '
7410 *P3
7420 OBJ1$="f" : OBJ2$="F"
7430 GOSUB *PICKUP
7440 RETURN
7600 '
7610 *P4
7620 CLS
7630 LOCATE 25,4 : INPUT "ｺｰﾄﾞ<END=999>";CD
7640 IF CD=999 THEN GOTO 7780
7650 GOSUB *DAT.SEARCH
7660 IF FOUND=-1 THEN 7600
7670 SEC2=FNSEC2(FOUND)
7680 REC2=FNREC2(FOUND)
7690 GET #2,SEC2
7700 CLS
7710 LOCATE 25,4 : PRINT NM$(REC2)
```

```
7720 LOCATE 25,6 : PRINT "トウロク シマスカ <y/n> ?"
7730 Y$=INKEY$ : IF Y$="" THEN GOTO 7730
7740 IF Y$="Y" OR Y$="y" THEN TOROKU=TOROKU+1
   : B(TOROKU)=FOUND : GOTO 7770
7750 IF Y$="N" OR Y$="n" THEN 7600
7760 BEEP : BEEP : GOTO 7730
7770 GOTO 7600
7780 NUMBER=TOROKU
7790 RETURN
8000 '
8010 *P5
8020 CLS
8030 LOCATE 25,4 : PRINT "■|    メニュー   |■"
8040 LOCATE 25,6 : PRINT "カイシャ ヤクイン・・・・・1"
8050 LOCATE 25,7 : PRINT "サラリーマン・・・・・・・・2"
8060 LOCATE 25,8 : PRINT "ガクセイ・・・・・・・・・3"
8070 LOCATE 25,9 : PRINT "シュフ・・・・・・・・・・・4"
8080 LOCATE 25,10: PRINT "ソノタ・・・・・・・・・・・5"
8090 LOCATE 25,12: INPUT "  input No.";OBJ$
8100 S.LENGTH=(TOTAL-1)¥50+2
8110 FOR I=2 TO S.LENGTH
8120  GET #1,I
8130  FOR J=1 TO 50
8140   PNT=PNT+1
8150   IF PROF$(PNT)=OBJ$ THEN GOTO 8160
        ELSE GOTO 8170
8160   TOROKU=TOROKU+1:B(TOROKU)=(I-2)*50+PNT
8170  NEXT J
8180  PNT=0
8190 NEXT I
8200 NUMBER=TOROKU
8210 RETURN
8400 '
8410 *P6
8420 CLS
8430 LOCATE 25,4 : PRINT "■|    メニュー   |■"
8440 LOCATE 25,6 : PRINT "30 サイ ミマン・・・・1"
8450 LOCATE 25,7 : PRINT "30 サイ イジョウ・・・2"
8460 LOCATE 25,9 : INPUT "  input No.";AGE
8470 S.LENGTH=(TOTAL-1)¥50+2
8480 FOR I=2 TO S.LENGTH
8490  GET #1,I
```

```
8500  FOR J=1 TO 50
8510   PNT=PNT+1
8520   IF AGE=2 THEN GOTO 8540
8530   IF CVI(OLD$(PNT))>0 AND CVI(OLD$(PNT))
        <30 THEN GOTO 8550 ELSE GOTO 8560
8540   IF CVI(OLD$(PNT))>=30 THEN GOTO 8550
        ELSE GOTO 8560
8550   TOROKU=TOROKU+1:B(TOROKU)=(I-2)*50+PNT
8560  NEXT J
8570  PNT=0
8580 NEXT I
8590 NUMBER=TOROKU
8600 RETURN
9000 '
9010 *PICKUP
9020 S.LENGTH=(TOTAL-1)¥50+2
9030 FOR I=2 TO S.LENGTH
9040  GET #1,I
9050  FOR J=1 TO 50
9060   PNT=PNT+1
9070   IF SEX$(PNT)=OBJ1$ OR SEX$(PNT)=OBJ2$
       THEN GOTO 9080 ELSE GOTO 9090
9080   TOROKU=TOROKU+1:B(TOROKU)=(I-2)*50+PNT
9090  NEXT J
9100  PNT=0
9110 NEXT I
9120 NUMBER=TOROKU
9130 RETURN
```

〈注〉このプログラムのサブルーチンで、該当するものがなく変数ＴＯＲＯＫＵの中味が０でリターンした場合(たとえば私たちの例では、職種別を選択し、かつ「その他」というメニューを選んだとき)、行番号670のＢ(Ｉ)＝０を満たしてしまうことになりＰ＝１となりますから、出力すべきレコードはないにもかかわらず、第１番目の人が必ず表示されてしまいます。この問題は、次の１行を追加しておけば解決できます。

```
485 IF NUMBER=0 THEN BEEP : LOCATE 30,4
: PRINT "ｶﾞｲﾄｳｼｬ ﾅｼ!" : FOR I=1 TO 2000
: NEXT I : GOTO *FCLOSE
```

2·7 メニュープログラム

それではこの章も、メニュープログラムを作って終りです。第1章のメニューと同じようなものですので、スクリーンエディタでステートメントの修正、追加をすると便利です。

変更点は、メニュープログラムのファイル名、キーインされたメニュー番号に対してRUNさせるファイル名などです。

第1章のメニューと区別するために、こちらは「No.2」としました。

キーインできましたら、このプログラムをセーブしてから実行してみてください。

```
10 ' file name ==> DMmenu.bas
20 ' 1984.1.20
30 ' Menu program for direct mail No.2
40 ' save "DMmenu.bas"
50 '
100 CONSOLE 0,25,0,0 : WIDTH 80,25
110 DEFINT A-Z : CLS
120 '
130 LOCATE 25,4
140 PRINT "■|ｱﾃﾅ ｶﾞｷ ﾒﾆｭｰ No.2|■"
150 LOCATE 25,6
160 PRINT "ﾃﾞｰﾀ ﾉ ｶｷｺﾐ･･････････････1"
170 LOCATE 25,7
180 PRINT "ﾃﾞｰﾀ ﾉ ｼｭｳｾｲ･･･････････2"
190 LOCATE 25,8
200 PRINT "ﾈﾝﾚｲ ﾉ ｺｳｼﾝ･････････････3"
210 LOCATE 25,9
220 PRINT "ｱﾃﾅ ｲﾝｻﾂ･･･････････････････4"
230 LOCATE 25,11
240 INPUT "    input number";NO
250 ON NO GOTO 270,280,290,300
260 '
270 RUN "DM02/1.bas"
280 RUN "DM02/2.bas"
290 RUN "DM02/3.bas"
300 RUN "DM02/4.bas"
```

第3章

通信文を漢字で印刷

3

3・1　漢字をプリンタで出力

この章では、前章の宛て名印刷プログラム

DM02/4.bas

に、さらにサブルーチンを追加して、右のページのように文書を日本語で出力するようにしてみたいと思います。

ＰＣ―8801mKⅡには「漢字ROM」が実装されていますので、さっそくこれを利用してみようというわけです。

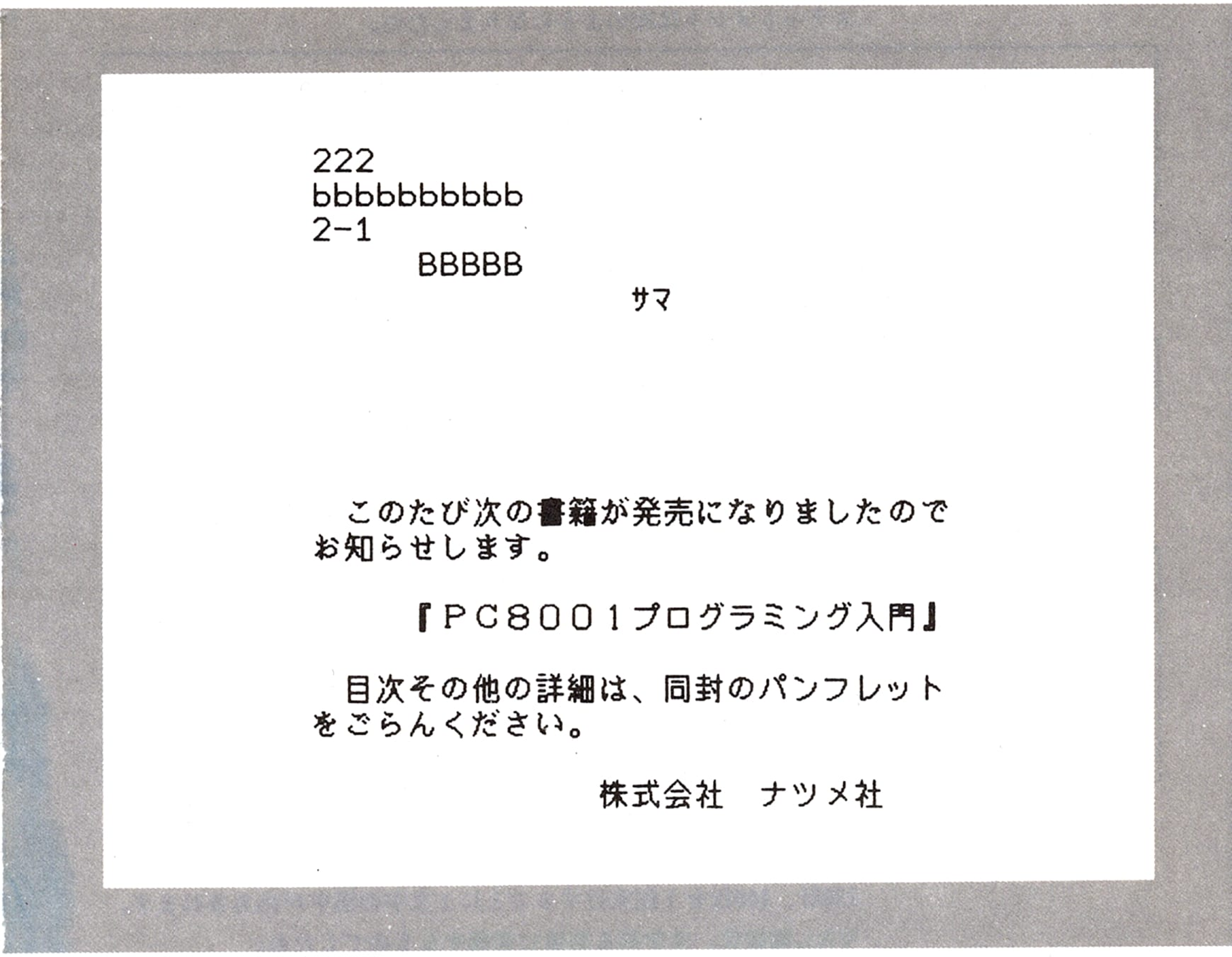
なお、プリンタにはＰＣ―8821またはＰＣ―8822が必要です。

漢字をプリンタに出力するには、『ユーザーズマニュアル』の巻末にある「漢字ＪＩＳコード表」により、出力したい漢字あるいはひらがななどを探し、その16進コードを求めます。そして、このコードと漢字出力コマンドの組み合わせによってプリンタで印刷することができるのです。

16進コードの求め方ですが、下に漢字ＪＩＳコード表の一部を引用し説明します。

たとえば、「入」という字の場合、表を左にたどると「4670」があり、上へたどると「E」があります。これらはいずれも16進表記です

		0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
ト	4540							兎	吐	堵	塗	妬	屠	徒	斗	杜	渡
	4550	登	菟	賭	途	都	鍍	砥	礪	努	度	土	奴	怒	倒	党	冬
	4560	凍	刀	唐	塔	塘	套	宕	島	嶋	悼	投	搭	東	桃	檮	棟
	4570	盗	淘	湯	濤	灯	燈	当	痘	禱	等	答	筒	糖	統	到	
	4620		董	蕩	藤	討	謄	豆	踏	逃	透	鐙	陶	頭	騰	闘	働
	4630	動	同	堂	導	憧	撞	洞	瞳	童	胴	萄	道	銅	峠	鴇	匿
	4640	得	徳	瀆	特	督	禿	篤	毒	独	読	栃	橡	凸	突	椴	届
	4650	鳶	苫	寅	酉	瀞	噸	屯	惇	敦	沌	豚	遁	頓	呑	曇	鈍
ナ	4660	奈	那	内	乍	凪	薙	謎	灘	捺	鍋	楢	馴	縄	畷	南	楠
	4670	軟	難	汝													
ニ	4670				二	尼	弐	邇	匂	賑	肉	虹	廿	日	乳	入	
	4720		如	尿	韮	任	妊	忍	認								


222
bbbbbbbbbb
2-1
BBBBB
サマ

このたび次の書籍が発売になりましたので
お知らせします。

『PC8001プログラミング入門』

目次その他の詳細は、同封のパンフレット
をごらんください。

株式会社　ナツメ社

が、最初の16進数の上３桁「467」と２つめの16進数「E」を組み合わせて「467E」としますと、これが「入」の16進のＪＩＳコードになります。

それでは、この16進コードを使ってどのようにプリンタに出するのか、簡単なプログラムを作って実際にやってみましょう。

出力する文章サンプルとして、

パソコン入門

を選びましょう。

いま説明した方法で、この６文字の16進コードを求め、まずデータ文を作ります。

データ文にするときに、たとえば「パ」の16進コードは「2551」ですから、２桁ずつに分け、「25」と「51」をコンマで区切って並べるようにします。

ステートメントは次のようになりましたね。

```
1000 DATA 25,51,25,3d,25,33,25,73,46,7e,4c,67
```


●DATA文の16進数は大文字でも小文字でもかまいません。

そしてこのデータをREAD文で読み、漢字に変換してプリンタへ出力するのですが、その前に、次のステートメントをキーインして、漢字出力のモードにしておかなければなりません。

```
100 LPRINT CHR$(27);"K";
```

これは公式ですから、理屈抜きに覚えてしまってください。

次にデータを読み込んで出力する部分です。

```
110 FOR I=1 TO 6
120 READ KJ1$,KJ2$
130 LPRINT CHR$(VAL("&H"+KJ1$));
140 LPRINT CHR$(VAL("&H"+KJ2$));
150 NEXT I
```

1000行のデータ文で漢字コードは２つで１組になっていますので、READ文でも２つずつ読み出すようにしました。

130行、140行を１回実行するごとに１文字の漢字が出力されます。

VAL関数は、文字列を数値に変換するものでしたね。

これも公式ですから覚えてしまってください。

すべてのデータを読み終えプリンタへ出力すると、FOR～NEXTループを抜け出てENDになるわけですが、このままですと漢字出力のモードが解除されていません。これを解除するには、他の印字方式のコードを送るようにします。たとえば、次のようなステートメントを追加すればよいでしょう。

```
160 LPRINT CHR$(27);"H";  ← ハイデンシティパイカ
170 LPRINT CHR$(&HF);     ← 拡大文字の解除
180 END
```

160行は、PC－8822の電源投入時の印字方式です。

それではプログラムをRUNさせてみてください。

「パソコン入門」と印刷されましたか？

3・2　文書のデータファイルを作る

さて、漢字出力の方法がわかったところで、簡単な文章のデータ文を作り、それを読み出してディスクに保存しておきましょう。

ディスクへの書き込みは、シーケンシャルファイルをオープンし、先ほどと同じように１バイトデータを２つずつREAD文で読み、それをＷＲＩＴＥ文でディスクへ落とします。

この部分のプログラムを次に示します。

前節のサンプルプログラムがメモリにまだ残っている人はＮＥＷコマンドで消してから入力してください。

```
10 ' file name ==> bunsho.bas
20 ' 1984.1.20
30 ' ｼｮｾｷ ﾉ ｱﾝﾅｲ
40 ' save "bunsho.bas"
50 '
100 OPEN "2:bunsho.dat" FOR OUTPUT AS #1
110 WHILE KJ1$<>"END"
120 READ KJ1$,KJ2$
130 WRITE #1,KJ1$,KJ2$
140 WEND
150 CLOSE
160 END
```

このプログラムのうしろにDATA文をつけ加えますが、最後にデータ末尾のマーク「ＥＮＤ」を入れておき、これを読んだときに書き込み終りとします。

それでは短い文章のデータを作りますが、文例として87ページの案内状を選ぶことにしましょう（159ページの図が出力例です）。

また、折り返しは左右20字で改行することにし、書名の前後を１行ずつあけることにします。これは便宜上20文字にしたのでして、漢字１文字はキャラクタ1.5字分に相当しますから、詰め込もうと思えば、１行50字くらいまで印刷できます。

漢字コードのDATA文を作るにあたって、スペースがどのように入り、どこで改行するか、もう一度サンプルの文章を書き出して整理してみましょう。

	△このたび	次の書籍が	発売になり	ましたので
	お知らせし	ます。		
1行あける →				
	△△△『P	C8001	プログラミ	ング入門』
1行あける →				
	△目次その	他の詳細は	、同封のパ	ンフレット
	をごらんく	ださい。		
1行あける →				
	△△△△△	△△△△株	式会社△ナ	ツメ社

↑ 5文字ずつ区切る

上の△印のところにはスペース（16進コード20）を入れます。

また、普通の改行のときは、行末にあまり使われない漢字コード、たとえば無限大のマーク「∞」の「2167」などを入れ、このコードを読んだときに改行かどうかの判定をします。

改行してさらに1行分あけたいときは、行末にやはりあまり使われない「∴」のコード「2168」などを入れておき、これによって処理を進めます。

上の図で、5文字ずつ区切りましたのは、DATA文1行について5文字分のコードをキーインするようにすると間違いをおかさないのでは、と思いこのようにしました。

あとは根気よく該当する日本語の漢字コードを探し入力していくのみです。

なお、漢字は音で索引します。たとえば「次」でしたら「ジ」ですから、漢字JISコード表の「シ」の欄を探せばよいですね。

私の入力したステートメントを右ページに示します。

1040行の行末に「21, 67」というデータがありますが、これが改行判定に使おうとするコードです。

それから、1080行末の「21, 68」、これは改行してさらに1行あける場合です。

また、1050行のように、その前の行までの漢字コードに対応する文章の一部をREM文として入れておくと、あとでなおしたりするとき便利です。

```
                スペース    こ     の     た     び
1000 '
1010 DATA 20,20,24,33,24,4e,24,3f,24,53
1020 DATA 3c,21,24,4e,3d,71,40,52,24,2c
1030 DATA 48,2f,47,64,24,4b,24,4a,24,6a
1040 DATA 24,5e,24,37,24,3f,24,4e,24,47,21,67
1050 ' ｺﾉﾀﾋﾞ ･･･････ﾅﾘﾏｼﾀ ﾉﾃﾞ
1060 ' line feed                    改行の判定に使います
1070 DATA 24,2a,43,4e,24,69,24,3b,24,37
1080 DATA 24,5e,24,39,21,23,21,68
1090 ' ｵｼﾗｾ ｼﾏｽ                     改行して、さらに1行分あけたい
1100 ' line feed two times          ときの判定用です
1110 DATA 20,20,20,20,20,20,21,58,23,50
1120 DATA 23,43,23,38,23,30,23,30,23,31
1130 DATA 25,57,25,6d,25,30,25,69,25,5f
1140 DATA 25,73,25,30,46,7e,4c,67,21,59,21,68
1150 ' ｢PC8001･････ﾆｭｳﾓﾝ
1160 ' line feed two times
1170 DATA 20,20,4c,5c,3c,21,24,3d,24,4e
1180 DATA 42,3e,24,4e,3e,5c,3a,59,24,4f
1190 DATA 21,22,46,31,49,75,24,4e,25,51
1200 DATA 25,73,25,55,25,6c,25,43,25,48,21,67
1210 ' ﾓｸｼﾞ･････････ﾊﾟﾝﾌﾚｯﾄ
1220 ' line feed
1230 DATA 24,72,24,34,24,69,24,73,24,2f
1240 DATA 24,40,24,35,24,24,21,23,21,68
1250 ' ｦ ｺﾞﾗﾝ ｸﾀﾞｻｲ
1260 ' line feed two times
1270 DATA 20,20,20,20,20,20,20,20,20,20
1280 DATA 20,20,20,20,20,20,20,20,33,74
1290 DATA 3c,30,32,71,3c,52,20,20,25,4a
1300 DATA 25,44,25,61,3c,52,END,END
1310 ' ･･････････････ﾅﾂﾒ ｼｬ
```

キーインできましたら、実行する前にまずこのプログラムをセーブしてください。

セーブできたら f・5 キーでRUNです。

コンマなどが落ちていると、あとで正しい出力結果が得られませんから注意してください。

3・3　漢字で通信文を

前節の文章書き込みプログラムの実行で、はたしてうまくいったかどうか、読み出して印刷してみましょう。

プログラムの構造は、160ページではREAD文でデータを読んでいましたが、今度はこれをシーケンシャルファイルの INPUT モード読み出せばよいのですから、みなさん、もうわかりますね。

プログラムを次に示します。

```
9500 '
9510 *BUNSHO
9520 OPEN "2:bunsho.dat" FOR INPUT AS #3
9530 LPRINT CHR$(27);"K";
9540 WHILE NOT(EOF(3))
9550 INPUT #3,KJ1$,KJ2$
9560 IF KJ1$="21" AND KJ2$="67" THEN LPRINT
      : GOTO 9600
9570 IF KJ1$="21" AND KJ2$="68" THEN LPRINT
      : LPRINT : GOTO 9600
9580 LPRINT CHR$(VAL("&H"+KJ1$));
9590 LPRINT CHR$(VAL("&H"+KJ2$));
9600 WEND
9610 CLOSE #3
9620 LPRINT CHR$(27);"H";
9630 LPRINT CHR$(&HF);
```

9530行で漢字出力モードにし、WHILE～WEND文を使ってファイル末尾までデータを読み出し、先ほど書き込んだ日本語の文章をプリンタに出力します。

9560行が改行の場合の処理です。この場合、読んだストリングデータは出力せずにWEND文へ飛ばしてしまいます。

9570行は改行してさらに1行あける場合ですので、LPRINT文を2つつけています。

キーインが終わりましたらRUNさせてみてください。159ページの宛て名を除く部分が出力されれば成功です。

もし、うまくいかないようでしたら、まず上のプログラムをチェックしてください。次に、このプログラムをとりあえず適当なファイル名でセーブし、今度は前節の書き込みプログラムをロードしてチェッ

クしてください。

さて、出力がうまくできましたら、

DM02/4.bas

にマージする作業をしますが、サブルーチン化しますので、次の2行を追加しておきます。

```
9640 FOR J=1 TO 10 : LPRINT : NEXT J
9650 RETURN
```

9640行は、次に続く宛て名との間をあけるための紙送り操作です。

それでは、この読み出し部分を次のように**アスキー形式でセーブ**しておきましょう。

```
save "bunmrg",a ↵
```

次に、**ベースとなるプログラムをロード**してください。

```
load "DM02/4.bas" ↵
```

そして、**マージ**です。

```
merge "bunmrg" ↵
```

これで2つのプログラムをドッキングさせる作業が終りました。リストコマンドで確認してみてください。

しかし、これではまだ漢字の文章は印刷されませんよ。もとの印刷ルーチンに、いまマージした文章出力サブルーチンへジャンプするステートメントがないからです。

2つのレコードごとに改ページすることにして、次のステートメントを追加します。

```
872    RC=RC+1
874    GOSUB *BUNSHO
876    IF RC=2 THEN RC=0 : LPRINT CHR$(12)
```

そして、プログラム先頭のREM文を変更し、新しいプログラムと

してセーブしておきましょう。

```
10 ' file name ==> DM02/5.bas
20 ' 1984.1.20
30 ' random data print out < KANJI >
40 ' save "DM02/5.bas"
50 '
```

```
save "DM02/5.bas"
```

このプログラムをRUNさせてためしてみてください。

また、あなたがたが実際に取り引きしている会社名などをディスクに書き込み、一方、簡単な案内状をも本章で学んだ方法で用意し、シミュレーションを行なってみるのもよいかと思います。

宛て名もランダムファイルを使って漢字出力をすることもできますが、コード入力になりますのでキーインの手間がかかります。ですから、宛て名はカタカナで表示し、先方に読んでいただく手紙のほうを漢字で印刷するようくふうしたほうが賢明でしょう。

ワープロ、ワープロとさわがれていますが、個人あるいは小さな会社で扱う文書はそれほど多くないと思います。いくつか「ひな型」を作っておいて、あとはDATA文の一部を変更して使えば、けっこう実務面で役立ちます。

市販のソフトを買っても、なかなか自分のニーズにぴったりのものが少ないようです。

この本で得た知識を使って、あなた専用の処理システムを作ってみてください。

3·4 マスターメニュープログラム

それでは最後に、マスターメニュープログラムを作っておきましょう。プログラムリストを次に示します。

```
10 ' file name ==> M
20 ' 1984.1.20
30 ' Master Menu program
40 ' save "M"
50 '
100 CONSOLE 0,25,0,0 : WIDTH 80,25
110 DEFINT A-Z : CLS
120 '
130 LOCATE 25,4
140 PRINT "■|  ﾏｽﾀｰ ﾒﾆｭｰ  |■"
150 LOCATE 25,6
160 PRINT "ｱﾃﾅ No.1･････････1"
170 LOCATE 25,7
180 PRINT "ｱﾃﾅ No.2･････････2"
190 LOCATE 25,8
200 PRINT "ｶﾝｼﾞ ﾌﾞﾝｼｮ･･･････3"
210 LOCATE 25,10
220 INPUT "   input number";NO
230 ON NO GOTO 250,260,270
240 '
250 RUN "DMMENU.bas"
260 RUN "DMmenu.bas"
270 RUN "DM02/5.bas"
```

プログラムファイル名は大文字の「M」と簡単にしました。マスターメニュープログラムさえ走らせてしまえば、あとはメニュー番号の入力だけで操作できるのですが、さらに楽をするためにキーインの回数も少ないほうがよいでしょう。

発売当初のシステムディスクには組み込まれていませんが、バージョンアップしたものには、オートスタートのユーティリティープログラム「setinf. n88」がありますから、これをRUNさせてIDセクタを書き換えておけば、電源を投入してシステムを立ち上げたときに、自動的にこのマスターメニュープログラムを実行させることができます。

セクタ書き換えの方法については消略しますが、むずかしくはあり

ませんから、ユーザーズマニュアルを読み、画面に表示されるメッセージに従ってキーインすれば、簡単に行なうことができます。

しかし、ファイル名を１文字にしておけば、キーインの手間もたいしてかかりませんし、時間的にも大差はないと思います。

何かほかのプログラムを修正したい場合などは、オートスタートで実行されたメニュープログラムをストップさせて、あらためてロードしなければならないのですから……。

おわりに

これで本書は終りです。

どうも長い間おつきあいありがとうございました。

表現力不足のために、わかりにくい箇所があったかもしれませんが、私としてはせいいっぱい解説したつもりです。

前著では、セクタを一定の型にあてはめた使い方を学びましたが、この本ではもう少し自由度をもたせましたので、あなたの実務への適用範囲が広がったのではないかと思います。

ディスクの扱いを身につけるには、まずランダムファイルの設計をマスターし、どのセクタのどのレコードをＧＥＴすればよいのかを実感することだと思います。

この辺のツボがつかめますと、あなたのディスクへの理解も深まり、と同時に、あなたがパソコンにやらせたいと思っていることとディスク管理との連絡がつき、急速に新しい視野がひらけてくると思います。

１つ残念なことは、もう１章設けて数値データを扱ってみたいと思ったのですが、年もおしせまり編集業務も忙しく執筆にばかりかかっていられないことと、編集長からはせきたてられるはで、一応の区切りもついたのでピリオドを打ちましたので、とうとう数値データはおあづけになってしまいました。

できることなら、少し複雑な数値データを扱った続刊も発行できたらと思っていますが、どうなることやら少々心配です。

この本をお読みいただいたみなさんから、読後感、あるいはご意見、ご希望など聞かせていただくと、今後の執筆の目やすにもなりますので、ぜひよろしくお願いいたします。宛て先は、「ナツメ社編集部パソコン係」でけっこうです。

どうもありがとうございました。

PC-8801mkII
データファイル入門2

編著者　ナツメ社編集部

発行者　田村正隆

発行所　株式会社　ナツメ社
東京都千代田区神田神保町1－52(〒101)
電話　03(291)1257(代表)
振替　東京3－58661
印　刷　ラン印刷社
製　本　三修紙工

ISBN4-8163-0366-9

PC-8801mkⅡ
データファイル入門[2]

発行――1985年4月15日
編著者――ナツメ社編集部
発行者――田村正隆
発行所――株式会社ナツメ社
郵便番号101
東京都千代田区神田神保町1-52
電話〈03〉291-1257
振替 東京3-58661
〈落丁・乱丁本はお取り替えします〉

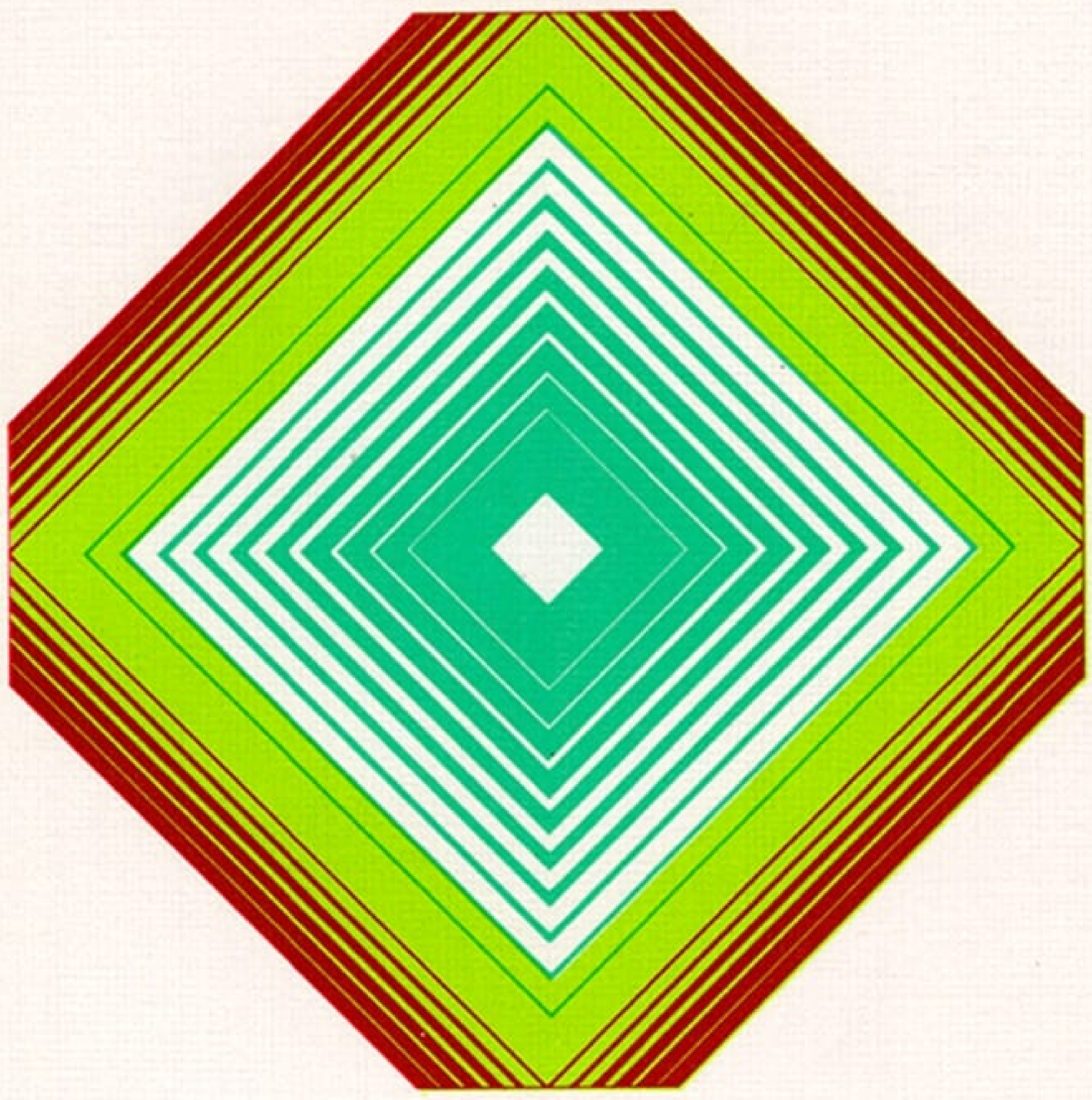
定価――1200円

PC-8801mkⅡ

データファイル入門[2]

ナツメ社
定価＝1200円

[目次より]



ISBN4-8163-0366-9 C2054 ¥1200E